Coalgebraic Components in a Many-Sorted Microcosm

Ichiro Hasuo[1,4], Chris Heunen[2], Bart Jacobs[2], and Ana Sokolova[3]

[1] RIMS, Kyoto University, Japan
[2] Radboud University Nijmegen, the Netherlands
[3] University of Salzburg, Austria
[4] PRESTO Research Promotion Program, Japan Science and Technology Agency

Abstract. The *microcosm principle*, advocated by Baez and Dolan and formalized for Lawvere theories lately by three of the authors, has been applied to coalgebras in order to describe compositional behavior systematically. Here we further illustrate the usefulness of the approach by extending it to a many-sorted setting. Then we can show that the coalgebraic component calculi of Barbosa are examples, with compositionality of behavior following from microcosm structure. The algebraic structure on these coalgebraic components corresponds to variants of Hughes' notion of *arrow*, introduced to organize computations in functional programming.

1 Introduction

Arguably the most effective countermeasure against today's growing complexity of computer systems is *modularity*: one should be able to derive the behavior of the total system from that of its constituent parts. Parts that were developed and tested in isolation can then safely be composed into bigger systems. Likewise, one would like to be able to prove statements about the compound system based on proofs of substatements about the parts. Therefore, the theoretical models should at the very least be such that their behavior is compositional.

This is easier said than done, especially in the presence of concurrency, that is, when systems can be composed in parallel as well as in sequence. The *microcosm principle* [1, 12] brings some order to the situation. Roughly speaking, compositionality means that the behavior of a compound system is the composition of the components' behaviors. The microcosm principle then observes that the very definition of composition of behaviors depends on composition of systems, providing an intrinsic link between the two.

The present article gives a rigorous analysis of compositionality of components as sketched above. Considering models as *coalgebras*, we study Barbosa's calculi of *components* [2, 3] as coalgebras with specified input and output interfaces. Explicitly, a component is a coalgebra for the endofunctor

$$F_{I,J} = (T(J \times \_))^I : \mathbf{Set} \to \mathbf{Set}, \tag{1}$$

where I is the set of possible input, and J that of output. The computational effect of the component is modeled by a monad T, as is customary in functional programming [25]. The monad T can capture features such as finite nondeterminism ($T = \mathcal{P}_\omega$), possible non-termination or exceptions ($T = 1 + \_$), probabilistic computation ($T = \mathcal{D}$), global states ($T = (S \times \_)^S$), or combinations of these.

To accommodate component calculi, the surrounding microcosm needs to be *many-sorted*. After all, composing components sequentially requires that the output of the first and the input of the second match up. This is elaborated on more precisely in §2. The contribution of the present article is twofold:

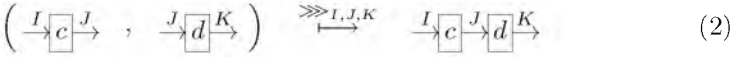
- a rigorous development of a many-sorted microcosm principle, in §4;
- an application of the many-sorted microcosm framework to component calculi, in §5.

It turns out that components as $F_{I,J}$-coalgebras carry algebraic structure that is a variant of Hughes' notion of *arrow* [14, 19].[5] Arrows, generalizing monads, have been used to model structured computations in semantics of functional programming. In §5 we give a rigorous proof that components indeed carry such arrow-like structure; however the calculation is overwhelming as it is. We shall exploit the fact that a Kleisli category $\mathcal{K}\ell(T)$, where the calculation takes place, also carries the same arrow-like structure. This allows us to use the axiomatization of the (shared) structure as an "internal language."

2 Leading example: sequential composition

We shall exhibit, using the following example, the kind of phenomena in component calculi that we are interested in.

For simplicity let us assume that we have no effect in components (i.e. $T = \mathrm{Id}, F_{I,J} = (J \times \_)^I$). Coalgebras for this functor are called *Mealy machines*, see e.g. [7]. A prominent operation in component calculi is *sequential composition*, or *pipeline*. It attaches two components with matching I/O interfaces, one after another:



(2)

Let X and Y be the state spaces of the components c and d, respectively. The resulting component $c \ggg_{I,J,K} d$ has the state space $X \times Y$;[6] first c produces output $j \in J$ that is fed into the input port of d. More precisely, we can define the coalgebra $c \ggg_{I,J,K} d$ to be the adjoint transpose of the following function.

$$I \times X \times Y \xrightarrow{\hat{c} \times d} J \times X \times (K \times Y)^J \xrightarrow{X \times \mathsf{ev}_J} K \times X \times Y \tag{3}$$

Here $\hat{c} : I \times X \to J \times X$ is the adjoint transpose of the coalgebra c, and $\mathsf{ev}_J : J \times (K \times Y)^J \to K \times Y$ is the obvious evaluation function.

[5] Throughout the paper the word "arrow" always refers to Hughes' notion. An "arrow" in a category (as opposed to an object) will be always called a *morphism*.

[6] We will use the infix notation for the operation $\ggg$. The symbol $\ggg$ is taken from that for (Hughes') arrows, whose relevance is explained in §5.

An important ingredient in the theory of coalgebra is "behavior-by-coinduction" [18]: when a state-based system is viewed as an F-coalgebra, then a *final* F-coalgebra (which very often exists) consists of all the "behaviors" of systems of type F. Moreover, the morphism induced by finality is the "behavior map": it carries a state of a system to its behavior. This view is also valid in the current example.

$$\begin{array}{ccc} FX & \dashrightarrow & FZ \\ {\scriptstyle c}\uparrow & & {\scriptstyle \text{final}}\uparrow\cong \\ X & \underset{\mathsf{beh}(c)}{\dashrightarrow} & Z \end{array}$$

A final $F_{I,J}$-coalgebra—where $F_{I,J} = (J \times \_)^I$—is carried by the set of stream functions $I^\omega \to J^\omega$ which are *causal*, meaning that the n-th letter of the output stream only depends on the first n letters of the input.[7] It conforms to our intuition: the "behavior" of such a component is what we see as an output stream when we feed it with an input stream. Let us denote the final $F_{I,J}$-coalgebra by

$$\zeta_{I,J} \;:\; Z_{I,J} \xrightarrow{\cong} F_{I,J}(Z_{I,J}) \;\;, \quad \text{that is,} \quad Z_{I,J} = \{t : I^\omega \to J^\omega \mid t \text{ is causal}\} \;\;.$$

The structure map $\zeta_{I,J}$ is described in detail in [29].

Then there naturally arises a "sequential composition" operation that is different from (2): it acts on *behaviors* of components, simply composing two behaviors of matching types.

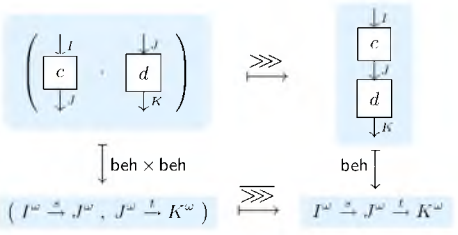
$$\begin{array}{rccccc} \overline{\ggg}_{I,J,K} \;: & Z_{I,J} & \times & Z_{J,K} & \longrightarrow & Z_{I,K} \\ & \big(I^\omega \xrightarrow{s} J^\omega & , & J^\omega \xrightarrow{t} K^\omega \big) & \longmapsto & I^\omega \xrightarrow{s} J^\omega \xrightarrow{t} K^\omega \end{array} \tag{4}$$

The following observation—regarding the two operations (2) and (4)—is crucial for our behavioral view on component calculi. The "inner" operation (4), although it naturally arises by looking at stream functions, is in fact induced by the "outer" operation (2). Specifically, it arises as the behavior map for the (outer) composition $\zeta_{I,J} \ggg_{I,J,K} \zeta_{J,K}$ of two final coalgebras.

$$\begin{array}{ccc} F_{I,K}(Z_{I,J} \times Z_{J,K}) & \dashrightarrow & F_{I,K}(Z_{I,K}) \\ {\scriptstyle \zeta_{I,J} \ggg \zeta_{J,K}}\uparrow & & {\scriptstyle \text{final}}\uparrow{\scriptstyle \zeta_{I,K}} \\ Z_{I,J} \times Z_{J,K} & \underset{\overline{\ggg}_{I,J,K}}{\dashrightarrow} & Z_{I,K} \end{array} \quad \text{i.e.} \quad \left(\begin{array}{c} \downarrow I \\ \boxed{\zeta_{I,J}} \\ \downarrow J \\ \boxed{\zeta_{J,K}} \\ \downarrow K \end{array} \right) \overset{\overline{\ggg}_{I,J,K}}{\dashrightarrow} \left(\begin{array}{c} \downarrow I \\ \boxed{\zeta_{I,K}} \\ \downarrow K \end{array} \right) \tag{5}$$

Note here that, due to our definition (3), the coalgebra $\zeta_{I,J} \ggg_{I,J,K} \zeta_{J,K}$ has a state space $Z_{I,J} \times Z_{J,K}$.

As to the two operations (2) and (4), we can ask a further question: are they compatible, in the sense that the the diagram on the right commute? One can think of this compatibility property as mathematical formulation of *compositionality*, a fundamental property in the theory of processes/components. The characterization of the inner operation by finality (5) is remarkably useful here; finality immediately yields a positive answer.



[7] This is how they are formalized in [29]. Equivalent formulations are: as string functions $I^* \to J^*$ that are length-preserving and prefix-closed [26]; and as functions $I^+ \to J$ where I^+ is the set of strings of length ≥ 1.

In fact, the *microcosm principle* is the mathematical structure that has been behind the story. It refers to the phenomenon that the same algebraic structure is carried by a category $\mathbb{C}$ and by an object $X \in \mathbb{C}$, a prototypical example being "a monoid object in a monoidal category" (see e.g. [24, §VII.3]). In [12] we presented another example eminent in the process theory: parallel composition of two coalgebras for the same signature functor, as well as parallel composition of their behaviors. Our story so far is yet another example taken from component calculi, with its new feature being that the algebraic structure is many-sorted.

3 FP-theory

3.1 Presenting algebraic structure as a category

Algebraic structure in this paper refers to the one in universal algebra (see e.g. [10]). We need it to be *many-sorted* in modeling component calculi. Algebraic structure consists of

- a set $\mathcal{S}$ of *sorts*;
- a set Σ of *operations*. Each operation $\sigma \in \Sigma$ is equipped with its *in-arity* $\mathsf{inar}(\sigma)$ given by a finite sequence of sorts denoted by $S_1 \times \cdots \times S_m$, and its *out-arity* $\mathsf{outar}(\sigma)$ that is some sort $S \in \mathcal{S}$;
- and a set E of equations.

A straightforward presentation of such is as a tuple $(\mathcal{S}, \Sigma, E)$ which is called an *algebraic specification* (see e.g. [17]).

In this paper we prefer different, categorical presentation of algebraic structure. The idea is that algebraic structure can be presented by a category $\mathbb{L}$ with:

- all the finite sequences of sorts $S_1 \times \cdots \times S_m$ as its objects;
- operations $\sigma \in \Sigma$ as morphisms $\mathsf{inar}(\sigma) \xrightarrow{\sigma} \mathsf{outar}(\sigma)$. Additionally, projections (such as $\pi_1 : S_1 \times S_2 \to S_1$) and diagonals (such as $\langle \mathrm{id}, \mathrm{id} \rangle : S \to S \times S$) are morphisms. So are (formal) products of two morphisms, equipping the category $\mathbb{L}$ with finite products. Besides we can compose morphisms in the category $\mathbb{L}$; that makes the morphisms in $\mathbb{L}$ precisely the *terms* composed using the operations in Σ;
- an equation as a commutative diagram. For example, when $\mathcal{S}$ is a singleton and we have a binary operation m, its associativity

$$x, y, z \vdash \mathsf{m}(x, \mathsf{m}(y, z)) = \mathsf{m}(\mathsf{m}(x, y), z) \quad \text{amounts to} \quad \begin{array}{ccc} 3 & \xrightarrow{\mathsf{m}\times\mathrm{id}} & 2 \\ {\scriptstyle \mathrm{id}\times\mathsf{m}}\downarrow & & \downarrow{\scriptstyle \mathsf{m}} \\ 2 & \xrightarrow[\mathsf{m}]{} & 1 \end{array} \quad . \tag{6}$$

See [17, §3.3] for the precise correspondence between an algebraic specification $(\mathcal{S}, \Sigma, E)$ and a category $\mathbb{L}$. The correspondence is not bijective; to be precise such a category $\mathbb{L}$ represents the *clone* of an algebraic specification (see e.g. [10]). Sketched in the above is the construction in one way, from $(\mathcal{S}, \Sigma, E)$ to $\mathbb{L}$.

In a one-sorted setting—where arities (objects of $\mathbb{L}$) are identified with natural numbers by taking their length—such a category $\mathbb{L}$ is called a *Lawvere theory* (see e.g. [12, 15, 22]). In a many-sorted setting, such a category $\mathbb{L}$—say a "many-sorted Lawvere theory"—is usually called a *finite-product theory*, or an *FP-theory*, see e.g. [4, 5].

Definition 3.1 (FP-theory) An *FP-theory* is a category with finite products.

The idea of such categorical presentation of algebraic structure originated in [22]. Significant about the approach is that one has a model as a functor.

Definition 3.2 (Set-theoretic model) Let $\mathbb{L}$ be an FP-theory. A *(set-theoretic) model* of $\mathbb{L}$ is a finite-product-preserving (*FP-preserving*) functor $X : \mathbb{L} \to \mathbf{Set}$ into the category **Set** of sets and functions.

Later in Def. 4.1 we introduce the notion of *category* with $\mathbb{L}$-structure—this is the kind of models of our interest—based on this standard definition.

To illustrate Def. 3.2 in a one-sorted setting, think about an operation $2 \xrightarrow{\mathsf{m}} 1$ which satisfies associativity (6). Let the image $X(1)$ of $1 \in \mathbb{L}$ be simply denoted by X; then $2 = 1 \times 1 \in \mathbb{L}$ must be mapped to the set X^2 by FP-preservation. By functoriality the morphism m is mapped to a morphism $X(\mathsf{m}) : X^2 \to X$ in **Set**, which we denote by $[\![\mathsf{m}]\!]_X$. This yields a binary operation on the set X. Moreover, the associativity diagram (6) in $\mathbb{L}$ is carried to a commutative diagram in **Set**; this expresses associativity of the interpretation $[\![\mathsf{m}]\!]_X$.

When $\mathbb{L}$ arises from a many-sorted algebraic specification, it is not a single set X that carries $\mathbb{L}$-structure; we have a family of sets $\{X(S)\}_{S \in \mathcal{S}}$—one for each sort S—as a carrier. By FP-preservation this extends to interpretation of products of sorts: $X(S_1 \times \cdots \times S_m) \cong X(S_1) \times \cdots \times X(S_m)$.[8] In this way an operation is interpreted with its desired domain and codomain.

3.2 The FP-theory PLTh

We now present a specific FP-theory which will be our working example. We list its sorts, operations and equations; these altogether induce an FP-theory in the way that we sketched above. We denote the resulting FP-theory by **PLTh**. Later in §5 we will see that this FP-theory represents Hughes' notion of *arrow*, without its first operation. One can think of **PLTh** as a basic component calculus modeling pipelines (PL for "pipeline").

Assumption 3.3 Throughout the rest of the paper we fix a base category $\mathbb{B}$ to be a Cartesian subcategory (i.e. closed under finite products) of **Set**. Its objects

[8] To be precise, one should see the right-hand side as denoting a specific choice of a product, say $(\cdots(XS_1 \times XS_2) \times \cdots) \times XS_m$. Because Cartesian products in **Set** are not strictly associative, one cannot force a functor X to be *strictly* FP-preserving. This is why the displayed equation holds only up-to isomorphism.

$I \in \mathbb{B}$ are sets that can play a role of an interface. Its morphisms $f : I \to J$—these are set-theoretic functions—represent "stateless" computations from I to J that can be realized by components with a single state.

The FP-theory **PLTh** is generated by:

- the sorts $\mathcal{S} = \{(I, J) \mid I, J \in \mathbb{B}\}$. Hence an object of **PLTh** can be written as a formal product $(I_1, J_1) \times \cdots \times (I_m, J_m)$. We denote the nullary product (i.e. the terminal object) by $1 \in \mathbf{PLTh}$;
- the operations:

$$\begin{array}{ll} \ggg_{I,J,K} \ : \ (I, J) \times (J, K) \longrightarrow (I, K) & \textit{sequential composition} \\ \mathsf{arr}\, f \ : \ 1 \longrightarrow (I, J) & \textit{pure function} \end{array}$$

  for each object $I, J, K \in \mathbb{B}$ and each morphism $f : I \to J$ in $\mathbb{B}$. Sequential composition is graphically understood as in (2). The component $\mathsf{arr}\, f$, intuitively, has a singleton as its state space and realizes "stateless" processing of data stream $\xrightarrow{I}\boxed{\mathsf{arr}\, f}\xrightarrow{J}$;
- the equations:
  - *associativity:*

$$a : (I, J), b : (J, K), c : (K, L) \ \vdash \ a \ggg (b \ggg c) = (a \ggg b) \ggg c \qquad (\ggg\text{-Assoc})$$

    for each $I, J, K, L \in \mathbb{B}$, omitting the obvious subscripts for $\ggg$, i.e.

$$\begin{array}{ccc} (I, J) \times (J, K) \times (K, L) & \xrightarrow{\ggg_{I,J,K} \times (K,L)} & (I, K) \times (K, L) \\ {\scriptstyle (I,J) \times \ggg_{J,K,L}}\downarrow & & \downarrow{\scriptstyle \ggg_{I,K,L}} \\ (I, J) \times (J, L) & \xrightarrow[\ggg_{I,J,L}]{} & (I, L) \end{array}$$

  - *preservation of composition:* for each composable pair of morphisms $f : I \to J$ and $g : J \to K$ in $\mathbb{B}$,

$$\emptyset \ \vdash \ \mathsf{arr}\,(g \circ f) = \mathsf{arr}\, f \ggg \mathsf{arr}\, g \qquad (\mathsf{arr}\text{-Func1})$$

    where $\emptyset$ denotes the empty context. That is as a diagram,

$$\begin{array}{ccc} 1 & \xrightarrow{\mathsf{arr}\, f \times \mathsf{arr}\, g} & (I, J) \times (J, K) \\ & {\scriptstyle \mathsf{arr}\,(g \circ f)}\searrow & \downarrow{\scriptstyle \ggg_{I,J,K}} \\ & & (I, K) \end{array}$$

  - *preservation of identities:* for each $I, J \in \mathbb{B}$,

$$a : (I, J) \ \vdash \ \mathsf{arr}\,\mathrm{id}_I \ggg_{I,I,J} a = a = a \ggg_{I,J,J} \mathsf{arr}\,\mathrm{id}_J \ . \qquad (\mathsf{arr}\text{-Func2})$$

For this FP-theory, the model of our interest is not a family of *sets* with this structure, but a family of *categories*, namely the category $\mathbf{Coalg}(F_{I,J})$ for each sort (I, J). Formalization of such an *outer model* carried by categories, together with that of an *inner model* carried by final coalgebras, is the main topic of the next section.

4 Microcosm model of an FP-theory

In this section we present our formalization of *microcosm models* for an FP-theory $\mathbb{L}$. It is about nested models of $\mathbb{L}$: the outer one ($\mathbb{L}$-*category*) being a family $\{\mathbb{C}(S)\}_{S\in\mathcal{S}}$ of categories; the inner one ($\mathbb{L}$-*object*) being a family $\{X_S \in \mathbb{C}(S)\}_{S\in\mathcal{S}}$ of objects. Its relevance has been mentioned in §2; we shall use our formalization to prove a general result (Thm. 4.7) that ensures compositionality.

In fact the formalization we present is essentially the one in our previous work [12]. Due to the space limit we cannot afford sufficient illustration of our seemingly complicated 2-categorical arguments. The reader is strongly suggested to have [12, §3] as her companion; the thesis [11, Chap. 5] of one of the authors has a more detailed account. What is new here, compared to [12], is the following.

- The algebraic structure of our interest is now many-sorted, generalizing $\mathbb{L}$ from a Lawvere theory to an FP-theory.
- Now we can accommodate categories with "pseudo" algebraic structure in our framework, such as monoidal categories as opposed to strict monoidal categories. We cannot avoid this issue in the current paper, where we deal with concrete models that satisfy equations only up-to isomorphisms.

4.1 Outer model: $\mathbb{L}$-category

Take the functor $F_{I,J} = (T(J \times \_))^I$, for which coalgebras are components (see §1). We would like that the categories $\{\mathbf{Coalg}(F_{I,J})\}_{I,J\in\mathbb{B}}$ model **PLTh**, the algebraic structure for pipelines in §3.2. That is, we need functors

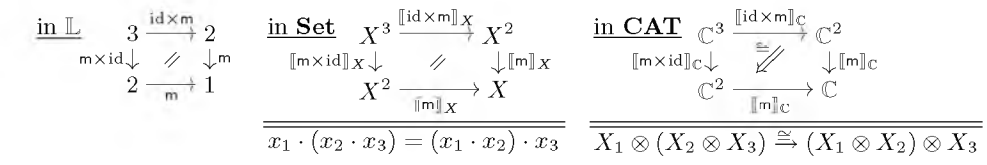
$$\begin{array}{lll} [\![\ggg_{I,J,K}]\!] & : & \mathbf{Coalg}(F_{I,J}) \times \mathbf{Coalg}(F_{J,K}) \to \mathbf{Coalg}(F_{I,K}) \quad \text{for each } I,J,K \in \mathbb{B}, \\ [\![\mathsf{arr}\, f]\!] & : & \mathbf{1} \to \mathbf{Coalg}(F_{I,J}) \qquad \text{for each morphism } f : I \to J \text{ in } \mathbb{B}, \end{array}$$

where **1** is a (chosen) terminal category, satisfying the three classes of equations of **PLTh** in §3.2. One gets pretty close to the desired definition of "category with $\mathbb{L}$-structure" by replacing "sets" by "categories" in Def. 3.2. That is, by having **CAT**—the 2-category of locally small categories, functors and natural transformations—in place of **Set**. In fact we did so in [12].

However here arises the problem of the right notion of "equality," as it always does when one moves up from n-categories to $(n+1)$-categories. In a *set* the right notion of "equality" is the identity between elements; this is why a monoid satisfies associativity up-to identity. In a *category* it is weakened into (coherent) isomorphisms between objects;[9] hence in a monoidal category multiplication is associative only up-to isomorphism. The definition of $\mathbb{L}$-categories must suitably address this issue. Specifically, an equation—a commutative diagram in $\mathbb{L}$—must now be carried to a diagram which is "commutative up-to isomorphism," i.e. a

[9] Equivalence of 0-cells is the right "equality" in a 2-category; biequivalence (see e.g. [27]) is the one in a 3-category; and so on.

diagram filled in with an iso-2-cell. Using the (one-sorted) example (6):



We have worked on the clue obtained in [12, §3.3] and [11, §5.3.3][10] and come to the following definition. In short, we get equations satisfied up-to isomorphism, by weakening a functor into a *pseudo functor*; the latter preserves identities and composition only up-to coherent isomorphisms (see e.g. [8]). The delicate question here is what it means for a pseudo functor to be *FP-preserving*. The conditions below are chosen so that Prop. 4.2 holds (see also Rem. 4.3 later).

Definition 4.1 (𝕃-category) An $\mathbb{L}$-*category* is a pseudo functor $\mathbb{C} : \mathbb{L} \to$ **CAT** that is *FP-preserving* in the following sense:[11]

1. the canonical map $\langle \mathbb{C}\pi_1, \mathbb{C}\pi_2 \rangle : \mathbb{C}(A_1 \times A_2) \to \mathbb{C}(A_1) \times \mathbb{C}(A_2)$ is an isomorphism for each $A_1, A_2 \in \mathbb{L}$;
2. the canonical map $\mathbb{C}(1) \to \mathbf{1}$ is an isomorphism;
3. it preserves identities up-to identity: $\mathbb{C}(\mathrm{id}) = \mathrm{id}$;
4. it preserves pre- and post-composition of identities up-to identity: $\mathbb{C}(\mathrm{id} \circ \mathsf{a}) = \mathbb{C}(a) = \mathbb{C}(\mathsf{a} \circ \mathrm{id})$;
5. it preserves composition of the form $\pi_i \circ \mathsf{a}$ up-to identity: $\mathbb{C}(\pi_i \circ \mathsf{a}) = \mathbb{C}(\pi_i) \circ \mathbb{C}(\mathsf{a})$. Here $\pi_i : A_1 \times A_2 \to A_i$ is a projection.

We shall often denote $\mathbb{C}$'s action $\mathbb{C}(\mathsf{a})$ on a morphism a by $[\![\mathsf{a}]\!]_{\mathbb{C}}$.

One consequence from the definition is that $\mathbb{C}$ also preserves composition of the form $\delta \circ \mathsf{a}$, where $\delta : A \to A \times A$ is a diagonal. It is illustrated in [11, 12] how pseudo functoriality induces isomorphisms up-to which equations are satisfied.

The definition is justified by the following fact. Its proof, as well as its generalization to other algebraic structure, is postponed to another venue.

Proposition 4.2 *Let us denote the Lawvere theory for monoids by* **MonTh**. *The 2-category* **MonCAT** *of monoidal categories, strong monoidal functors and monoidal transformations is equivalent to the 2-category of* **MonTh**-*categories with suitable 1- and 2-cells.* □

In Def. 4.1 one can replace **CAT** by any 2-category with finite 2-products and obtain a more general notion of pseudo $\mathbb{L}$-model. Such generality is not needed in this paper.

[10] Later we came to know that the idea is folklore at least for "monoidal" theories. It is mentioned in [13] as *Segalic* presentation of monoidal categories.

[11] To be precise, each of the conditions 3–5 means that the corresponding mediating isomorphism (as part of the definition of a pseudo functor) is actually the identity.

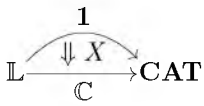
Remark 4.3 A standard way to avoid the complication with pseudo algebraic structure is by a coherence result [20, 24]. For example: every monoidal category is equivalent to a strict one. This, however, only gives us a biequivalence (see e.g. [27]) between **MonCAT** and the 2-category of strict monoidal categories. Although one readily sees that the latter is equivalent to the 2-category of strict **MonTh**-categories, the two correspondence results combined only yield biequivalence. In contrast, Prop. 4.2 realizes *equivalence* between **MonCAT** and the 2-category of **MonTh**-categories, by fine-tuning the latter notion.

Remark 4.4 In [21] a different approach for modeling pseudo algebraic structure is presented. There a "Lawvere theory" for monoidal categories is a 2-category with all the coherent isomorphisms α, λ, ρ explicit as 2-cells. This allows one to have a model as a (strict) 2-functor. In contrast, in our approach, monoids and monoidal categories are specified by the same Lawvere theory **MonTh** without any 2-cells; the former is an FP-functor into **Set** and the latter a pseudo FP-functor into **CAT**.

4.2 Inner model: $\mathbb{L}$-object

Once we have an outer model $\mathbb{C}$ of $\mathbb{L}$, we can define the notion of *inner model in* $\mathbb{C}$. It is a family of objects $\{X_S \in \mathbb{C}(S)\}_{S \in \mathcal{S}}$ which carries $\mathbb{L}$-structure in the same way as a monoid object in a monoidal category carries structure as a monoid [24, §VII.3]. Its relevance to component calculi is explained in §2 where final coalgebras carry an inner model and realize composition of behaviors.

Definition 4.5 ($\mathbb{L}$-object) Let $\mathbb{C} : \mathbb{L} \to \mathbf{CAT}$ be an $\mathbb{L}$-category. An $\mathbb{L}$-*object* in $\mathbb{C}$ is a lax natural transformation (see e.g. [8])

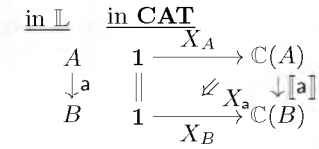


which is *FP-preserving* in the sense that: it is strictly natural with regard to projections and diagonals (see [12, Def. 3.4]). Here $\mathbf{1} : \mathbb{L} \to \mathbf{CAT}$ denotes the constant functor to a (chosen) terminal category $\mathbf{1}$.

An $\mathbb{L}$-object is also called a *microcosm model* of $\mathbb{L}$, emphasizing that it is a model that resides in another model $\mathbb{C}$.

The definition is abstract and it might be hard to grasp how it works. While the reader is referred to [11, 12] for its illustration, we shall point out its highlights.

An $\mathbb{L}$-object X, as a lax natural transformation, consists of the following data:



- its components $X_A : \mathbf{1} \to \mathbb{C}(A)$, identified with objects $X_A \in \mathbb{C}(A)$, for each $A \in \mathbb{L}$;
- mediating 2-cells X_{a}, as shown on the right, for each morphism a in $\mathbb{L}$.

Generalizing the illustration in [11, 12] one immediately sees that

- X's components are determined by those $\{X_S\}_{S\in\mathcal{S}}$ for sorts. The latter extend to an object $S_1 \times \cdots \times S_m$ by:

$$\mathbb{C}(S_1 \times \cdots \times S_m) \ni X_{S_1\times\cdots\times S_m} \overset{\cong}{\longmapsto} (X_{S_1}, \ldots, X_{S_m}) \in \mathbb{C}(S_1) \times \cdots \times \mathbb{C}(S_m) \ ;$$

- an operation σ is interpreted on X by means of the mediating 2-cell X_σ;
- equations hold due to the coherence condition on the mediating 2-cells.

4.3 Categorical compositionality

Here we shall present a main technical result, namely the compositionality theorem (Thm. 4.7). It is a straightforward many-sorted adaptation of [12, Thm. 3.9], to which we refer for its proof and more illustration.

Definition 4.6 (𝕃-functor) Let $\mathbb{C}, \mathbb{D}$ be $\mathbb{L}$-categories. A *lax $\mathbb{L}$-functor* $F : \mathbb{C} \to \mathbb{D}$ is a lax natural transformation $\mathbb{L} \overset{\mathbb{C}}{\underset{\mathbb{D}}{\Downarrow F}} \mathbf{CAT}$ that is FP-preserving in the same sense as in Def. 4.5. Similarly, a *strict $\mathbb{L}$-functor* is a strict natural transformation of the same type.

A lax/strict $\mathbb{L}$-functor determines, as its components, a family of functors $\{F_A : \mathbb{C}(A) \to \mathbb{D}(A)\}_{A\in\mathbb{L}}$. Much like the case for an $\mathbb{L}$-object, it is determined by the components $\{F_S : \mathbb{C}(S) \to \mathbb{D}(S)\}_{S\in\mathcal{S}}$ on sorts.

Theorem 4.7 (Compositionality) *Let $\mathbb{C}$ be an $\mathbb{L}$-category, and $F : \mathbb{C} \to \mathbb{C}$ be a lax $\mathbb{L}$-functor. Assume further that there is a final coalgebra $\zeta_A : Z_A \overset{\cong}{\to} F_A(Z_A)$ for each $A \in \mathbb{L}$.*

1. *The family $\{\mathbf{Coalg}(F_A)\}_{A\in\mathbb{L}}$ carries an $\mathbb{L}$-category.*
2. *The family $\{\zeta_A \in \mathbf{Coalg}(F_A)\}_{A\in\mathbb{L}}$ carries a microcosm model of $\mathbb{L}$.*
3. *The family $\{\mathbb{C}(A)/Z_A\}_{A\in\mathbb{L}}$ of slice categories carries an $\mathbb{L}$-category.*
4. *The family of functors $\{\mathsf{beh}_A : \mathbf{Coalg}(F_A) \to \mathbb{C}(A)/Z_A\}_{A\in\mathbb{L}}$, where*

$$\mathsf{beh}_A \ : \ \mathbf{Coalg}(F_A) \longrightarrow \mathbb{C}(A)/Z_A \quad \textit{is by coinduction} \quad \begin{array}{ccc} F_A X & \dashrightarrow & F_A(Z_A) \\ {\scriptstyle c}\uparrow & & {\scriptstyle final}\uparrow\cong \\ X & \underset{\mathsf{beh}_A(c)}{\dashrightarrow} & Z_A \end{array}$$

is a strict $\mathbb{L}$-functor. □

An informal reading of the theorem is as follows. To get a "nice" interpretation of a component calculus $\mathbb{L}$ by F-coalgebras, it suffices to check that

- the base category $\mathbb{C}$ models $\mathbb{L}$, and
- the functor F is "lax-compatible" with $\mathbb{L}$.

These data interpret $\mathbb{L}$ on the category of coalgebras, yielding composition of components (the point 1.). Final coalgebras acquire canonical inner $\mathbb{L}$-structure, yielding composition of behaviors (the point 2.). Finally, relating the two interpretations, compositionality is guaranteed (the point 4.).

5 Taxonomy of FP-theories for component calculi

Up to now we have kept an FP-theory $\mathbb{L}$ as a parameter and have developed a uniform framework that applies to any $\mathbb{L}$. Now we start looking at: concrete models (components as coalgebras); and three concrete FP-theories **PLTh**, **ArrTh** and **MArrTh** that express basic component calculi. The latter two are equipped with different "parallel composition" operations.

Notably the algebraic structure expressed by **ArrTh** is that of (Hughes') *arrow* [14], equivalently that of *Freyd categories* [23], the notions introduced for modeling structured computations in functional programming.

The main result in this section is that the categories $\{\mathbf{Coalg}(F_{I,J})\}_{I,J}$—modeling components with $F_{I,J} = (T(J \times \_))^I$—carry **ArrTh**-structure. If additionally the effect monad T is commutative, then $\{\mathbf{Coalg}(F_{I,J})\}_{I,J}$ a forteriori carries stronger **MArrTh**-structure. These results parallel classic results in categorical semantics of functional programming, investigating (pre)monoidal structure of a Kleisli category.

5.1 The FP-theories ArrTh, MArrTh

We shall add, to the FP-theory **PLTh** in §3.2, a suitable "parallel composition" operation and equational axioms to obtain the FP-theory **ArrTh**. By imposing stronger equational axioms we get the FP-theory **MArrTh**.

In **ArrTh** one has additional *sideline* operations

$$\mathsf{first}_{I,J,K} \;:\; (I,J) \longrightarrow (I \times K, J \times K) \;, \quad \text{graphically} \quad \begin{array}{c} \downarrow I \\ \boxed{a} \\ \downarrow J \end{array} \;\overset{\mathsf{first}_{I,J,K}}{\longmapsto}\; \left(\begin{array}{c|c} \begin{array}{c} \downarrow I \\ \boxed{a} \\ \downarrow J \end{array} & \begin{array}{c} K \\ \downarrow \\ K \end{array} \end{array} \right)$$

for each $I, J, K \in \mathbb{B}$. The equations regarding these are:

$$\mathsf{first}\, a \ggg \mathsf{arr}\, \pi = \mathsf{arr}\, \pi \ggg a \qquad (\rho\text{-NAT})$$

$$\mathsf{first}\, a \ggg \mathsf{arr}(\mathrm{id} \times f) = \mathsf{arr}(\mathrm{id} \times f) \ggg \mathsf{first}\, a \qquad (\mathsf{arr}\text{-CENTR})$$

$$\mathsf{first}\, a \ggg \mathsf{arr}\, \alpha = \mathsf{arr}\, \alpha \ggg \mathsf{first}(\mathsf{first}\, a) \qquad (\alpha\text{-NAT})$$

$$\mathsf{first}(\mathsf{arr}\, f) = \mathsf{arr}(f \times \mathrm{id}) \qquad (\mathsf{arr}\text{-PREMON})$$

$$\mathsf{first}(a \ggg b) = \mathsf{first}\, a \ggg \mathsf{first}\, b \qquad (\mathsf{first}\text{-FUNC})$$

It is easy to recover the omitted subscripts for arr, $\ggg$ and first. In the equations, f denotes a morphism in the base category $\mathbb{B}$. In (ρ-NAT), if a has the type (I, J) then the π on the left is the projection $\pi : J \times 1 \overset{\cong}{\to} J$ in $\mathbb{B}$. In (α-NAT), α's are associativity isomorphisms like $I \times (J \times K) \overset{\cong}{\to} (I \times J) \times K$ in $\mathbb{B}$.

Remark 5.1 In fact the equation (ρ-NAT) holds for any projection $\pi : J \times K \to J$ without requiring $K = 1$; one can derive this general case from the special case and other equations. However, the special case has a clearer role in the corresponding premonoidal structure (see §5.2). Namely, it is the naturality requirement of the right-unit isomorphism $\rho_J = \mathsf{arr}\, \pi_J$ with $\pi_J : J \times 1 \overset{\cong}{\to} J$.

In **MArrTh**, instead of the operations first, one has

$$\|_{I,J,K,L} : (I,J) \times (K,L) \longrightarrow (I \times K, J \times L) \qquad \textit{synchronous composition}$$

for each $I, J, K, L \in \mathbb{B}$. The equations are:

$$(a \parallel b) \ggg (c \parallel d) = (a \ggg c) \parallel (b \ggg d) \qquad (\|\text{-Func1})$$
$$\mathsf{arr}\,\mathrm{id} \parallel \mathsf{arr}\,\mathrm{id} = \mathsf{arr}\,\mathrm{id} \qquad (\|\text{-Func2})$$
$$a \parallel (b \parallel c) \ggg \mathsf{arr}\,\alpha = \mathsf{arr}\,\alpha \ggg (a \parallel b) \parallel c \qquad (\alpha\text{-Nat})$$
$$(a \parallel \mathsf{arr}\,\mathrm{id}_1) \ggg \mathsf{arr}\,\pi = \mathsf{arr}\,\pi \ggg a \qquad (\rho\text{-Nat})$$
$$\mathsf{arr}(f \times g) = \mathsf{arr}\,f \parallel \mathsf{arr}\,g \qquad (\mathsf{arr}\text{-Mon})$$
$$(a \parallel b) \ggg \mathsf{arr}\,\gamma = \mathsf{arr}\,\gamma \ggg (b \parallel a) \qquad (\gamma\text{-Nat})$$

Here α's are associativity isomorphisms, and π's are projections like $J \times 1 \stackrel{\cong}{\to} J$, as in **ArrTh**. The morphisms γ in (γ-Nat) are symmetry isomorphisms like $J \times I \stackrel{\cong}{\to} I \times J$ in $\mathbb{B}$. One readily derives, from (ρ-Nat) and (γ-Nat), the equation

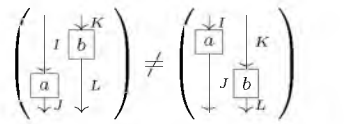
$$(\mathsf{arr}\,\mathrm{id}_1 \parallel a) \ggg \mathsf{arr}\,\pi' = \mathsf{arr}\,\pi' \ggg a \qquad (\lambda\text{-Nat})$$

where π''s are (second) projections like $1 \times I \stackrel{\cong}{\to} I$.

The reason that we have the first operation in **ArrTh**, instead of $\|$ as in **MArrTh**, should be noted. The first operation in **ArrTh** yields the operation

$$\mathsf{second}_{I,J,K} : (I,J) \longrightarrow (K \times I, K \times J) \quad \text{by } \mathsf{second}\,a = \mathsf{arr}\,\gamma \ggg \mathsf{first}\,a \ggg \mathsf{arr}\,\gamma \ ,$$

where γ's are symmetry isomorphisms. But the equations in **ArrTh** do not derive $\mathsf{second}\,b \ggg \mathsf{first}\,a = \mathsf{first}\,a \ggg \mathsf{second}\,b$; that is, the two systems on the right should not be identified. Indeed there are many situations where these two systems are distinct. Assume that we have the global state monad $T = (S \times \_)^S$ as effect in $F_{I,J}$. One can think of a global state $s \in S$ as residing in the ambience of components, unlike local/internal states that are inside components (i.e. coalgebras). When a component executes, it changes the current global state as well as its internal state; hence the order of execution of a and b does matter. In contrast, when one interprets **MArrTh** in $\{\mathbf{Coalg}(F_{I,J})\}_{I,J}$, the natural axiom ($\|$-Func1) requires the above two systems to be equal.



5.2 Set-theoretic models: arrows, Freyd categories

The FP-theories **PLTh**, **ArrTh** and **MArrTh** and their set-theoretic models are, in fact, closely related with some notions that have been studied extensively in semantics of functional programming. Here we elaborate on the relationship; it will be exploited later in §5.3.

To start with, **ArrTh** is almost exactly a categorical presentation of the axiomatization of Hughes' *arrow* [14] (specifically the axiomatization in [19]),

the only gap being the one explained in Remark 5.1. The notion of arrow generalizes that of *monad* (modeling effects, i.e. structured output [25, 31]) and that of *comonad* (modeling structured input [30]); the notion of arrow models "structured computations" in general. See e.g. [19, §2.3].

Definition 5.2 An *arrow* is a set-theoretic model (Def. 3.2) of **ArrTh**.

It had been folklore, and was proved in [19], that an arrow is the same thing as a *Freyd category*. A Freyd category is a symmetric premonoidal category $\mathbb{K}$ together with a Cartesian category $\mathbb{B}$ embedded via an identity-on-object strict premonoidal functor $\mathbb{B} \to \mathbb{K}$, subject to a condition on center morphisms. The notion is introduced in [28], and it is named as such later in [23].

In this way one can look at **ArrTh** as an axiomatization of the notion of Freyd category. There is a similar corresponding structure for the (stronger) FP-theory **MArrTh**, which explains its name **MArrTh**.

Definition 5.3 Let $\mathbb{B}$ be a Cartesian (hence monoidal) category. A *monoidal Freyd category* on $\mathbb{B}$ is a symmetric monoidal category $\mathbb{K}$ together with a strictly monoidal, identity-on-object functor $\mathbb{B} \to \mathbb{K}$.

Proposition 5.4 *1. A set-theoretic model* $\mathbf{ArrTh} \to \mathbf{Set}$ *of* **ArrTh** *is the same thing as a Freyd category.*

2. A set-theoretic model $\mathbf{MArrTh} \to \mathbf{Set}$ *of* **MArrTh** *is the same thing as a monoidal Freyd category.*

Proof. The first point is simply the correspondence result [19, Thm. 6.1] between arrows and Freyd categories, put together with Def. 4.5. The proof of the second point goes similar. □

Remark 5.5 To be precise, the correspondences in the previous proposition are *equivalences* of suitable categories. This is for the same reason as the category of set-theoretic models of the Lawvere theory **MonTh** is *equivalent* to the category **Mon** of monoids. These correspondences fail to be *isomorphisms* because of the possible different choices of products in **Set**.

The notions of (monoidal) Freyd category were introduced in [28], prior to arrows, as axiomatizations of the structure possessed by a Kleisli category $\mathcal{K}\ell(T)$ for a monad T. Here a Kleisli category is understood as a category of types and effectful computations [25]. The results [28, Cor. 4.2 & 4.3]—showing that $\mathcal{K}\ell(T)$ indeed induces a Freyd category—now read as follows, in view of Prop. 5.4. For the notion of strong/commutative monad, see e.g. [16, §3].

Proposition 5.6 *Let* $\mathbb{B}$ *be our base category (see Assumption 3.3).*

1. *A monad* T *on* **Set** *induces a model* $\mathcal{K}\ell(T) : \mathbf{ArrTh} \to \mathbf{Set}$. *Specifically, its carrier set* $\mathcal{K}\ell(T)(I, J)$ *for the sort* (I, J) *is the homset* $\mathrm{Hom}_{\mathcal{K}\ell(T)}(I, J)$; arr *is interpreted by the Kleisli inclusion functor;* $\ggg$ *is by composition in* $\mathcal{K}\ell(T)$*; and* first *is obtained using the canonical strength* st *of* T. *Recall that every monad on* **Set** *is strong.*

2. *Furthermore, if T is commutative then it induces a model $\mathcal{K}\ell(T)$ of* **MArrTh**. *The operation $\|$ is interpreted using T's double strength.* □

Thanks to the proposition we know that all the equations in **ArrTh** or in **MArrTh** hold in $\mathcal{K}\ell(T)$, with the operations suitably interpreted. This fact will be heavily exploited in the equational reasoning later in §5.3.

For **PLTh**—the parallel-free part of **ArrTh** and **MArrTh**—a set-theoretic model is an arrow without first, or equivalently, a category $\mathbb{K}$ with an identity-on-object functor $\mathbb{B} \to \mathbb{K}$. Yet another characterization of this structure, discovered in [19], is as a monoid object in the monoidal category of bifunctors $[\mathbb{B}^{\mathrm{op}} \times \mathbb{B}, \mathbf{Set}]$ where the monoidal products are given by the so-called *Day tensor* [9]. Equivalently, it is a monad on $\mathbb{B}$ in the bicategory of profunctors (also called distributors or bimodules, see e.g. [6, 8]).

5.3 PLTh, ArrTh and MArrTh as component calculi

We now show that our coalgebraic modeling of components indeed models the calculi **PLTh**, **ArrTh** and **MArrTh**, according to the choice of an effect monad T. The result parallels Prop. 5.6.

Throughout the rest of the section we denote by $\mathbb{L}_{\mathrm{CC}}$ any one of **PLTh**, **ArrTh** and **MArrTh** (CC for "component calculus").

In view of Thm. 4.7, we only need to establish that: 1) the (constant) map $(I, J) \mapsto \mathbf{Set}$ extends to an $\mathbb{L}_{\mathrm{CC}}$-category; and 2) $\{F_{I,J} : \mathbf{Set} \to \mathbf{Set}\}_{I,J}$ extends to a lax $\mathbb{L}_{\mathrm{CC}}$-functor. Then Thm 4.7 ensures that the components (as coalgebras) and their behaviors (as elements of final coalgebras) carry a microcosm model of $\mathbb{L}_{\mathrm{CC}}$, and that compositionality holds.

For 1), we interpret the operations in the following way. The guiding question is: what is the state space of the resulting component, when we apply an operation to components as its arguments.

Definition 5.7 We denote by **Set** the $\mathbb{L}_{\mathrm{CC}}$-category defined as follows. It maps each sort (I, J) to $\mathbf{Set} \in \mathbf{CAT}$; and it interprets operations by

$$\begin{array}{ccccccccc} \mathbf{1} & \overset{[\![\mathsf{arr}\ f]\!]}{\longrightarrow} & \mathbf{Set}, & \mathbf{Set} & \overset{[\![\mathsf{first}]\!]}{\longrightarrow} & \mathbf{Set}, & \mathbf{Set} \times \mathbf{Set} & \overset{[\![\ggg]\!]}{\longrightarrow} & \mathbf{Set}, \\ * & \longmapsto & 1 & X & \longmapsto & X & (X, Y) & \longmapsto & X \times Y \end{array} \qquad \begin{array}{ccccc} \mathbf{Set} \times \mathbf{Set} & \overset{[\![\|]\!]}{\longrightarrow} & \mathbf{Set}. \\ (X, Y) & \longmapsto & X \times Y \end{array}$$

We are using the same notation **Set** for models of three different FP-theories. This will not cause confusion.

Lemma 5.8 *The data in Def. 5.7 indeed determine FP-preserving pseudo functors. In particular, all the equations in* **PLTh**, **ArrTh** *and* **MArrTh** *are satisfied up-to coherent isomorphisms.*

Proof. Easy. The equations hold only up-to isomorphisms because, for example, the associativity $X \times (Y \times U) \cong (X \times Y) \times U$ in **Set** is only an isomorphism. □

The requirement 2)—that $\{F_{I,J} : \mathbf{Set} \to \mathbf{Set}\}_{I,J \in \mathbb{B}}$ extends to a lax $\mathbb{L}_{\mathrm{CC}}$-functor—puts additional demands on T. This is parallel to Prop. 5.6. Still the actual calculation is overwhelming. We notice that all the operations that appear throughout the calculation can be described as morphisms in the Kleisli category $\mathcal{K}\ell(T)$. Therefore, by Prop. 5.6, they are themselves subject to the equations in $\mathbb{L}_{\mathrm{CC}}$. This substantially simplifies the calculation.

Lemma 5.9 *For the endofunctors $F_{I,J} = (T(J \times \_))^I : \mathbf{Set} \to \mathbf{Set}$, the following hold.*

1. *The family $\{F_{I,J} : \mathbf{Set} \to \mathbf{Set}\}_{I,J}$ extends to a lax* **ArrTh***-functor* $\mathbf{Set} \to \mathbf{Set}$. *Therefore so it does to a lax* **PLTh***-functor.*
2. *If T is commutative, $\{F_{I,J} : \mathbf{Set} \to \mathbf{Set}\}_{I,J}$ extends to a lax* **MArrTh***-functor* $\mathbf{Set} \to \mathbf{Set}$.

Proof. What we need to do is: first to "interpret operations" on $\{F_{I,J}\}_{I,J}$; second to check if these interpreted operations "satisfy equations."

More specifically, to make $\{F_{I,J}\}_{I,J}$ into a lax $\mathbb{L}_{\mathrm{CC}}$-functor, we need to have a mediating 2-cell corresponding to each operation (such as $\ggg$). For example:

$$\begin{array}{ll} \underline{\text{in } \mathbb{L}_{\mathrm{CC}}} & \underline{\text{in } \mathbf{CAT}} \\ (I,J) \times (J,K) & \mathbf{Set} \times \mathbf{Set} \xrightarrow{F_{I,J} \times F_{J,K}} \mathbf{Set} \times \mathbf{Set} \\ \downarrow \ggg & [\![\ggg]\!] = \times \downarrow \quad \swarrow F_{\ggg} \quad \downarrow [\![\ggg]\!] = \times \\ (I,K) & \mathbf{Set} \xrightarrow[F_{I,K}]{} \mathbf{Set} \end{array} \tag{7}$$

In the diagram, we have denoted the binary product in **Set** by the boldface $\times : \mathbf{Set} \times \mathbf{Set} \to \mathbf{Set}$ to distinguish it from the binary product $\times$ in **CAT**. The needed 2-cell $F_{\ggg}$ is nothing but a natural transformation

$$F_{\ggg} \ : \ F_{I,J}X \times F_{J,K}Y \longrightarrow F_{I,K}(X \times Y) \ . \tag{8}$$

After that we have to show that these mediating 2-cells satisfy the coherence condition. In the current setting where the domain category $\mathbb{L}_{\mathrm{CC}}$ is syntactically generated by sorts, operations and equations, it amounts to checking if the mediating 2-cells "satisfy the equations." Taking ($\ggg$-Assoc) as an example, it means showing the following equality between 2-cells.

$$\begin{array}{ccc} \times \circ (\times \times \mathrm{id}) & & \times \circ (\mathrm{id} \times \times) \\ \mathbf{Set}^3 & \xrightarrow{F_{I,J} \times F_{J,K} \times F_{K,L}} & \mathbf{Set}^3 \\ {\scriptstyle \mathrm{id} \times \times} \downarrow & \swarrow {\scriptstyle \mathrm{id} \times F_{\ggg}} & \downarrow {\scriptstyle \mathrm{id} \times \times} \\ \mathbf{Set}^2 & \xrightarrow{F_{I,K} \times F_{K,L}} & \mathbf{Set}^2 \\ {\scriptstyle \times} \downarrow & \swarrow {\scriptstyle F_{\ggg}} & \downarrow {\scriptstyle \times} \\ \mathbf{Set} & \xrightarrow[F_{I,L}]{} & \mathbf{Set} \end{array} \quad = \quad \begin{array}{ccc} & & \\ \mathbf{Set}^3 & \xrightarrow{F_{I,J} \times F_{J,K} \times F_{K,L}} & \mathbf{Set}^3 \\ {\scriptstyle \times \times \mathrm{id}} \downarrow & \swarrow {\scriptstyle F_{\ggg} \times \mathrm{id}} & \downarrow {\scriptstyle \times \times \mathrm{id}} \\ \mathbf{Set}^2 & \xrightarrow{F_{I,K} \times F_{K,L}} & \mathbf{Set}^2 \\ {\scriptstyle \times} \downarrow & \swarrow {\scriptstyle F_{\ggg}} & \downarrow {\scriptstyle \times} \\ \mathbf{Set} & \xrightarrow[F_{I,L}]{} & \mathbf{Set} \end{array} \tag{9}$$

Here α denotes the associativity isomorphism $X \times (Y \times U) \xrightarrow{\cong} (X \times Y) \times U$. See [11, Rem. 5.4.1] for more illustration. One readily sees that the equation (9)

boils down to commutativity of the following diagram in **Set**.

$$\begin{array}{ccccc} F_{I,J}X \times (F_{J,K}Y \times F_{K,L}U) & \xrightarrow{\mathrm{id} \times F_{\ggg}} & F_{I,J}X \times F_{J,L}(Y \times U) & \xrightarrow{F_{\ggg}} & F_{I,L}(X \times (Y \times U)) \\ \alpha\downarrow & & & & \downarrow F_{I,L}\alpha \\ (F_{I,J}X \times F_{J,K}Y) \times F_{K,L}U & \xrightarrow[F_{\ggg} \times \mathrm{id}]{} & F_{I,K}(X \times Y) \times F_{K,L}U & \xrightarrow[F_{\ggg}]{} & F_{I,L}((X \times Y) \times U) \end{array} \tag{10}$$

To summarize: we shall introduce a natural transformation F_σ, for each operation σ, like in (8); and check if they satisfy all the equations like in (10). While the first task is straightforward, the second is painfully complicated as it is. We shall only do the aforementioned examples in $\mathbb{L}_{\mathrm{CC}}$ for demonstration.

In the sequel, let us denote the set-theoretic model of $\mathbb{L}_{\mathrm{CC}}$ induced by the Kleisli category (in Prop. 5.6) by $\mathcal{K}\ell(T)$. In particular, we have $\mathcal{K}\ell(T)(I, J) = (TJ)^I$. Hence the natural transformation $F_{\ggg}$ in (8) is of the type

$$\mathcal{K}\ell(T)(I, J \times X) \times \mathcal{K}\ell(T)(J, K \times Y) \to \mathcal{K}\ell(T)(I, K \times (X \times Y)) \ .$$

We define it to be the following composition of morphisms in $\mathbb{L}_{\mathrm{CC}}$, interpreted in $\mathcal{K}\ell(T)$.

$$\begin{array}{c} (I, J \times X) \times (J, K \times Y) \overset{\mathrm{id} \times \mathsf{first}}{\longrightarrow} (I, J \times X) \times \big(J \times X,\, (K \times Y) \times X\big) \\ \overset{\ggg}{\longrightarrow} \big(I,\, (K \times Y) \times X\big) \overset{(\_) \ggg (\mathsf{arr}\, \alpha^{-1})}{\longrightarrow} (I, K \times (X \times Y)) \end{array} \tag{11}$$

One can readily come up with such a morphism in $\mathbb{L}_{\mathrm{CC}}$ for each of the other operations.

Let us prove that $F_{\ggg}$ thus defined satisfies (10). The morphisms in the diagram (10) can be also written as morphisms in $\mathbb{L}_{\mathrm{CC}}$, interpreted in $\mathcal{K}\ell(T)$. Hence we shall do the equational reasoning in $\mathbb{L}_{\mathrm{CC}}$, using the equational axioms in §5.1. To reduce the number of parentheses, the terms are presented modulo associativity ($\ggg$-Assoc) of $\ggg$. The letters c, d and e are variables of sorts $(I, J \times X)$, $(J, K \times Y)$ and $(K, L \times U)$, respectively.

$$\begin{array}{lll} & \text{(The path, first down, then to the right)} & \\ = & c \ggg \mathsf{first}\, d \ggg \mathsf{arr}\, \alpha^{-1} \ggg \mathsf{first}\, e \ggg \mathsf{arr}\, \alpha^{-1} & \\ = & c \ggg \mathsf{first}\, d \ggg \mathsf{first}\, \mathsf{first}\, e \ggg \mathsf{arr}\, \alpha^{-1} \ggg \mathsf{arr}\, \alpha^{-1} & (\alpha\text{-Nat}) \\ = & c \ggg \mathsf{first}\, d \ggg \mathsf{first}\, \mathsf{first}\, e \ggg \mathsf{arr}\, (\alpha^{-1} \circ \alpha^{-1}) & (\mathsf{arr}\text{-Func1}) \\ = & c \ggg \mathsf{first}\, d \ggg \mathsf{first}\, \mathsf{first}\, e \ggg \mathsf{arr}\, \big((L \times \alpha) \circ \alpha^{-1} \circ (\alpha^{-1} \times U)\big) & (\dagger) \\ = & c \ggg \mathsf{first}\, d \ggg \mathsf{first}\, \mathsf{first}\, e \ggg \mathsf{arr}\, (\alpha^{-1} \times U) \ggg \mathsf{arr}\, \alpha^{-1} \ggg \mathsf{arr}\, (L \times \alpha) & (\mathsf{arr}\text{-Func1}) \\ = & c \ggg \mathsf{first}\, d \ggg \mathsf{first}\, \mathsf{first}\, e \ggg \mathsf{first}\, (\mathsf{arr}\, \alpha^{-1}) \ggg \mathsf{arr}\, \alpha^{-1} \ggg \mathsf{arr}\, (L \times \alpha) & (\mathsf{first}\text{-Premon}) \\ = & c \ggg \mathsf{first}\big((d \ggg \mathsf{first}\, e) \ggg \mathsf{arr}\, \alpha^{-1}\big) \ggg \mathsf{arr}\, \alpha^{-1} \ggg \mathsf{arr}(L \times \alpha) & (\mathsf{first}\text{-Func}) \\ = & \text{(The path, first to the right, then down)} & \end{array}$$

Here the equality (†) is because of the "pentagon" coherence for α in **Set**. Naturality of $F_{\ggg}$ in X, Y, as well as satisfaction of the other equations, can be derived by similar calculation. □

For obtaining a microcosm model we need final coalgebras. This depends on the "size" of the monad T; all the examples of T listed in §1, except for $T = \mathcal{D}$, satisfy the requirement.

Theorem 5.10 *Assume that, for each $I, J \in \mathbb{B}$, we have a final coalgebra $\zeta_{I,J} : Z_{I,J} \xrightarrow{\cong} F_{I,J}(Z_{I,J})$.*

1. *The family $\{\mathbf{Coalg}(F_{I,J})\}_{I,J}$ forms an* **ArrTh***-category, so a* **PLTh***-category.*
2. *The family $\{Z_{I,J} \in \mathbf{Set}\}_{I,J}$ forms an* **ArrTh***-object, hence a* **PLTh***-object. Compositionality holds in the sense of Thm. 4.7.4.*
3. *If T is commutative, then the above two are also an* **MArrTh***-category and an* **MArrTh***-object, respectively.*

Proof. By Thm. 4.7, Lem. 5.8 and Lem. 5.9. □

6 Conclusions and Future Work

We have extended our previous formalization of the microcosm principle [12] to a many-sorted setting. This allowed to include Barbosa's component calculi [2] as examples. We studied three concrete calculi that are variants of the axiomatization of arrows, demonstrating similarity between components and structured computations.

As future work, we are interested in further extensions of the component calculi that allow modeling of further interesting examples like wiring and merging components, queues, stacks, and folders of stacks with feedback, etc., as presented in [2]. The proof methods that we derived from the microcosm framework will be useful in its course.

On the more abstract side, it is interesting to elevate the arguments in §5 further, to the bicategory **DIST** of distributors, with **CAT** embedded in it via the Yoneda embedding. Such a higher level view on the matter might reveal further microcosm instances in our proof methods.

Acknowledgments Thanks are due to Kazuyuki Asada, Masahito Hasegawa, Paul-Andrě Melliès and John Power for helpful discussions and comments.

References

1. J.C. Baez and J. Dolan. Higher dimensional algebra III: n-categories and the algebra of opetopes. *Adv. Math*, 135:145–206, 1998.
2. L.S. Barbosa. Towards a calculus of state-based software components. *Journ. of Universal Comp. Sci.*, 9(8):891–909, 2003.
3. L. Barbosa. *Components as Coalgebras.* PhD thesis, Univ. Minho, 2001.
4. M. Barr and C. Wells. *Toposes, Triples and Theories.* Springer, Berlin, 1985. Available online.
5. M. Barr and C. Wells. *Category Theory for Computing Science.* Centre de recherches mathématiques, Université de Montréal, 3rd edn., 1999.
6. J. Bénabou. Distributors at work. Lecture notes by Thomas Streicher, 2000. `www.mathematik.tu-darmstadt.de/~streicher/FIBR/DiWo.pdf.gz`.
7. M.M. Bonsangue, J.J.M.M. Rutten and A. Silva. Coalgebraic logic and synthesis of Mealy machines. In R.M. Amadio, editor, *FoSSaCS*, vol. 4962 of *Lect. Notes Comp. Sci.*, pp. 231–245. Springer, 2008.

8. F. Borceux. *Handbook of Categorical Algebra*, vol. 50, 51 and 52 of *Encyclopedia of Mathematics*. Cambridge Univ. Press, 1994.
9. B.J. Day. On closed categories of functors. In *Reports of the Midwest Category Seminar IV*, vol. 137 of *Lect. Notes Math.*, pp. 1–38. Springer-Verlag, 1970.
10. K. Denecke and S.L. Wismath. *Universal Algebra and Applications in Theoretical Computer Science*. Chapman & Hall, 2002.
11. I. Hasuo. *Tracing Anonymity with Coalgebras*. PhD thesis, Radboud University Nijmegen, 2008.
12. I. Hasuo, B. Jacobs and A. Sokolova. The microcosm principle and concurrency in coalgebra. In *Foundations of Software Science and Computation Structures*, vol. 4962 of *Lect. Notes Comp. Sci.*, pp. 246–260. Springer-Verlag, 2008.
13. C. Hermida. A roadmap to the unification of weak categorical structures: transformations and equivalences among the various notions of pseudo-algebra, 2004. `maggie.cs.queensu.ca/chermida/papers/roadmap.pdf`.
14. J. Hughes. Generalising monads to arrows. *Science of Comput. Progr.*, 37(1–3):67–111, 2000.
15. M. Hyland and J. Power. The category theoretic understanding of universal algebra: Lawvere theories and monads. *Elect. Notes in Theor. Comp. Sci.*, 172:437–458, 2007.
16. B. Jacobs. Semantics of weakening and contraction. *Ann. Pure & Appl. Logic*, 69(1):73–106, 1994.
17. B. Jacobs. *Categorical Logic and Type Theory*. North Holland, Amsterdam, 1999.
18. B. Jacobs and J.J.M.M. Rutten. A tutorial on (co)algebras and (co)induction. *EATCS Bulletin*, 62:222–259, 1997.
19. B. Jacobs, C. Heunen and I. Hasuo. Categorical semantics for arrows. *Journ. Funct. Progr.*, 2009. To appear.
20. A. Joyal and R. Street. Braided tensor categories. *Adv. Math*, 102:20–78, 1993.
21. S. Lack and J. Power. Lawvere 2-theories. Presented at CT2007, 2007. `www.mat.uc.pt/~categ/ct2007/slides/lack.pdf`.
22. F.W. Lawvere. *Functorial Semantics of Algebraic Theories and Some Algebraic Problems in the Context of Functorial Semantics of Algebraic Theories*. PhD thesis, Columbia University, 1963. Reprints in Theory and Applications of Categories, 5 (2004) 1–121.
23. P.B. Levy, A.J. Power and H. Thielecke. Modelling environments in call-by-value programming languages. *Inf. & Comp.*, 185(2):182–210, 2003.
24. S. Mac Lane. *Categories for the Working Mathematician*. Springer, Berlin, 2nd edn., 1998.
25. E. Moggi. Notions of computation and monads. *Inf. & Comp.*, 93(1):55–92, 1991.
26. D. Pattinson. An introduction to the theory of coalgebras. Course notes for NASSLLI, 2003. `www.indiana.edu/~nasslli`.
27. A.J. Power. Why tricategories? *Inf. & Comp.*, 120(2):251–262, 1995.
28. J. Power and E. Robinson. Premonoidal categories and notions of computation. *Math. Struct. in Comp. Sci.*, 7(5):453–468, 1997.
29. J.J.M.M. Rutten. Algebraic specification and coalgebraic synthesis of Mealy automata. *Elect. Notes in Theor. Comp. Sci.*, 160:305–319, 2006.
30. T. Uustalu and V. Vene. Comonadic notions of computation. *Elect. Notes in Theor. Comp. Sci.*, 203(5):263–284, 2008.
31. P. Wadler. Monads for functional programming. In *Marktoberdorf Summer School on Program Design Calculi*. Springer-Verlag, 1992.