Interaction Laws of Monads and Comonads

Shin-ya Katsumata
National Institute of Informatics
Tokyo, Japan
s-katsumata@nii.ac.jp

Exequiel Rivas
Inria
Paris, France
exequiel.rivas-gadda@inria.fr

Tarmo Uustalu
Reykjavik University
Reykjavik, Iceland
Tallinn University of Technology
Tallinn, Estonia
tarmo@ru.is

Abstract

We introduce and study functor-functor and monad-comonad interaction laws as mathematical objects to describe interaction of effectful computations with behaviors of effect-performing machines. Monad-comonad interaction laws are monoid objects of the monoidal category of functor-functor interaction laws. We show that, for suitable generalizations of the concepts of dual and Sweedler dual, the greatest functor resp. monad interacting with a given functor or comonad is its dual while the greatest comonad interacting with a given monad is its Sweedler dual. We relate monad-comonad interaction laws to stateful runners. We show that functor-functor interaction laws are Chu spaces over the category of endofunctors taken with the Day convolution monoidal structure. Hasegawa's glueing endows the category of these Chu spaces with a monoidal structure whose monoid objects are monad-comonad interaction laws.

CCS Concepts: • **Theory of computation → Functional constructs**; **Program semantics**.

Keywords: effectful computation, monads, comonads, interaction laws, dual of a functor, Sweedler dual of a monad, Chu spaces, Hasegawa's glueing

ACM Reference Format:
Shin-ya Katsumata, Exequiel Rivas, and Tarmo Uustalu. 2020. Interaction Laws of Monads and Comonads. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS '20), July 8–11, 2020, Saarbrücken, Germany.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3373718.3394808

1 Introduction

What does it mean to run an effectful program, abstracted into a computation?

In this paper, we take the view that an effectful computation does not perform its effects (we mean the effects surviving any internal handling). These effects are to be provided externally. The computation can only proceed if placed in an environment that can provide its effects, e.g, respond to the computation's requests for input, listen to its output, resolve its nondeterministic choices by tossing a coin, consistently respond to its fetch and store commands. Abstractly, such an environment is a machine whose implementation is opaque to us; we can witness its behavior, its evolution through externally visible states. It is useful to think of the computation as a client depending on an effect service and of the machine or its behavior as a server; they can work together following some agreed protocol.

To formalize this intuition, we follow Moggi [26] in regards to allowed computations (the chosen notions of computation) and describe them using a monad T on the category of types and functions that we want to compute on. (In Plotkin and Power's approach [30], they are described with a Lawvere theory.) Allowed machine behaviors (the chosen notion of machine behavior), at the same time, are described with a comonad D. An operational semantics is then described by what we call an *interaction law*, a natural transformation $\psi : TX \times DY \to X \times Y$ compatible with the (co)unit and (co)multiplication. This polymorphic function sends a computation (TX) and a machine behavior from some initial state (DY) into a return value X and a final state Y. Compatibility with the unit and counit means that interaction of a "just returning" computation with a machine behavior must terminate immediately in its initial state. Compatibility with multiplication and comultiplication means that interaction of a sequence of computations amounts to a sequence of interactions whereby the second interaction starts from the state where the first finished. As we explore these interactions, it will became evident that it is also fine to work with notions of computation and machine behavior that do not include "just returning" or/and are not closed under sequential composition; those can be described with plain functors instead of a monad and a comonad.

To illustrate how monad-comonad interaction laws describe execution of a program, we propose a simple example. Suppose we want to model computations that issue input and output requests. The arities of the these operations can

LICS '20, July 8–11, 2020, Saarbrücken, Germany

ACM ISBN 978-1-4503-7104-9/20/07…$15.00
https://doi.org/10.1145/3373718.3394808

be packed into a functor $F\,Z = (I \Rightarrow Z) + (O \times Z)$. Computations over a set of values X that use these operations are elements of the carrier $F^*\,X = \mu Z.\, X + F\,Z$ of the free algebra of F on X; the functor F^* underlies the free monad on F. Concretely, these computations are wellfounded trees whose nodes correspond to either input or output requests. A node for an input request has I many children; a node for output request has one child, but is labelled by an element of O. Leaves correspond to return points, they are labelled by elements from the set of values X.

Next we might wonder in which environments these computations can be expected to run. We need something that provides at least input and output services (but it may be able to do more). A natural choice is to use a coalgebra of the functor $G\,W = (I \times W) \times (O \Rightarrow W)$. This is a state machine operating on a set of states Y according to a dynamics $\gamma : Y \to G\,Y$ that can supply the computation an input or accept its output when so requested. It should be clear that any such state machine determines a polymorphic function $\theta_X : F^*X \times Y \to X \times Y$, which works by walking up the given computation tree from the root until a leaf is reached, using the state machine started in the given initial state as an oracle. At an input request node, the machine provides a token from I determining the child node to go to and transitions to a new state. At an output request node, there is no choice about the child node in the computation tree, but the token from O labelling the current node determines the next state of the machine. When a leaf is reached, the leaf label is returned together with the current state of the machine that has therewith become final. Uustalu [41] called such a function θ a *stateful runner* of F^*.

But we can go a little more abstract. Given a state machine, i.e., a set of states Y and dynamics $\gamma : Y \to GY$, any initial state $y \in Y$ determines a machine behavior (an abstraction collecting all runs of the machine from this initial state) as an element of the carrier $G^\dagger\,Y = \nu W.\, Y \times GW$ of the cofree coalgebra of G on Y; we note that $G^\dagger$ underlies the cofree comonad on G. A machine behavior like this is a non-wellfounded tree whose nodes are labelled with states from Y. Every node's first child is additionally labelled with tokens from I, but it also has O many further children with no additional label. To execute computations, machine behaviors are enough. There is an evident function $\psi_{X,Y} : F^*X \times G^\dagger Y \to X \times Y$ describing how any computation runs against any given machine behavior; the return value and final state are determined by traversing the computation and behavior trees simultaneously in a lock-step fashion until a leaf is reached in the computation tree. This function, polymorphic not only in the set of values X but also the set of states Y, is an example of a *monad-comonad interaction law* between F^* and $G^\dagger$.

This interaction law is not incidental, it is canonical. The comonad $G^\dagger$ is in a special relationship to the monad F^*. In a good technical sense it is the "greatest" comonad interacting with F^*. This turns out to be a consequence of the functor G being the "greatest" functor interacting with the functor F.

In this paper, we find out some basic properties of monad-comonad interaction laws, some constructions of more sophisticated monad-comonad interactions from simpler ones etc. We take special interest in the questions (a) which is the "greatest" comonad interacting with the given monad T (so any interaction law of T with any comonad would factor through the canonical interaction law of T with this comonad)? and (b) which is the "greatest" monad (resp. functor) interacting with a given comonad D (or functor G)? To answer these, we draw inspiration from algebra, where the dual of a vector space V is $V^\circ = V \to \mathbb{K}$. The answer to (b) turns out to be: the dual of D (resp. G), under a suitably generalized concept of dual. Question (a) is harder. To answer it, we need to generalize the concept of what is called the Sweedler dual. The greatest comonad interacting with T is the Sweedler dual of T.

The contributions in this paper are the following:

(i) We introduce *functor-functor interaction laws*, define the dual of a functor, and show that the greatest functor interacting with a given functor is its dual (Section 2).
(ii) We study *monad-comonad interaction laws* as monoid objects of the category of functor-functor interaction laws. We show that the dual lifts from functors to comonads and that the greatest monad interacting with a given comonad is its dual whereas for monads it does not lift like this; for the greatest comonad interacting with a monad, the Sweedler dual is needed (Section 3).
(iii) We relate monad-comonad interaction laws to *stateful runners* (Section 4).
(iv) Using the Day convolution and duoidal categories, we recast monad-comonad interaction laws as monoid-comonoid interaction laws, and relate them to two standard constructions: Chu spaces and Hasegawa's glueing (Section 5). This gives us a method for computing the Sweedler duals of free monoids (monads) and their quotients by equations.

We consider a number of examples of interaction laws. We calculate the dual of a functor and Sweedler dual of a monad in a number of cases, but we also demonstrate that sometimes the dual of a functor or the Sweedler dual of a monad is either completely or somewhat degenerate, which has the consequence that any interaction laws and runners are then necessarily trivial or limited. (The simplest of those cases is when the functor comes with a nullary operation, which means that there is an exception effect.) This is not a defect in the definitions of interaction laws or runners. It is a manifestation of the fact that not every denotationally motivated notion of computation admits operational semantics that rely on state only. A remedy that is both mathematically and

practically well-motivated is provided by *residual* functor-functor interaction laws, monad-comonad interaction laws and stateful runners as generalizations where the machine need not be able to perform all effects of the computation, some residual effects can remain after interaction or running (see the end of Section 2).[1]

Plotkin and Pretnar's effect handlers [33] are a mechanism for treating effects inside an effectful program. On the theoretical level, handlers are a straightforward thing—they are just monad algebras or models—but they are a very useful concept practically. The novelty of this paper lies in a new perspective on the scenario where a program's effects are performed by an external machine in an interaction. In the categorical semantics literature, this scenario has been studied in a small number of papers on comodels (see Section 6 for a brief review). We contend that the phenomena surrounding this kind of interaction admit elegant explanations in terms of constructions developed elsewhere in mathematics. One of the points we would like to make is that it is useful to consider not only canonical interactions between a notion of computation and its dual, but also general interactions.

We assume the reader to be familiar with adjunctions / monads / comonads, extensive categories [12], Cartesian closed categories, ends/coends (the end-coend calculus [10, 23]). In a nutshell, extensive categories are categories with well-behaved finite coproducts.

Throughout most of the paper (Sections 2–4), we work with one fixed base category C that we assume to be extensive with finite products. For some constructions (the dual of a functor), we also need that C is Cartesian closed. For the same constructions, we also use certain ends that we either explicitly show to exist or only use when they happen to exist. We also rely on Cartesian closedness in most examples.

2 Functor-functor interaction

We begin with functor-functor interaction, to then proceed to the monad-comonad interaction laws in the next section.

2.1 Functor-functor interaction laws

In a functor-functor interaction law, computations over a set of values X are elements of FX where F is a given functor fixing the notion of computation considered. Machine behaviors over a set of states Y are elements of GY where G is another given functor. Any allowed computation and any allowed machine behavior can help each other reach a return value and a final state by interacting as prescribed.

We define an *functor-functor interaction law* on C to be given by two endofunctors F, G together with a family of maps

$$\phi_{X,Y} : FX \times GY \to X \times Y$$

natural in X and Y.

[1] The details can be found in the full version of this paper [22, Sec.5].

Example 2.1. The archetypical example of a functor-functor interaction law is defined by $FX = A \Rightarrow X$, $GY = A \times Y$, and $\phi\,(f,(a,y)) = (f\,a, y)$ for some fixed object A. But we can also take, e.g., $FX = A \Rightarrow X$, $GY = C \times Y$, and $\phi\,(f,(c,y)) = (f\,(h\,c), y)$ for some fixed map $h : C \to A$.

Example 2.2. A more interesting example is obtained by taking $FX = A \Rightarrow (B \times X)$, $GY = A \times (B \Rightarrow Y)$, $\phi\,(f,(a,g)) =$ let $(b,x) \leftarrow f\,a$ in $(x, g\,b)$. We can vary this by taking $GY = (A \Rightarrow B) \Rightarrow (A \times Y)$ and $\phi\,(f,h) =$ let $\langle f_0, f_1\rangle \leftarrow f; (a,y) \leftarrow h\,f_0$ in $(f_1 a, y)$.

Example 2.3. If C has the relevant initial algebras and final coalgebras, we can get interaction laws by iterating the above interactions, e.g., with $FX = \mu Z.\, X + (A \Rightarrow (B \times Z))$ and $GY = \nu W.\, Y \times (A \times (B \Rightarrow W))$. We will shortly explain the construction of ϕ in this case.

An *interaction law map* between (F,G,ϕ), (F',G',ϕ') is given by natural transformations $f : F \to F'$, $g : G' \to G$ such that

$$\begin{array}{ccccc} & {\scriptstyle \mathrm{id}\times g_Y} & FX \times GY & \xrightarrow{\phi_{X,Y}} & X \times Y \\ FX \times G'Y & \nearrow & & & \| \\ & \searrow & & & \| \\ & {\scriptstyle f_X \times \mathrm{id}} & F'X \times G'Y & \xrightarrow{\phi'_{X,Y}} & X \times Y \end{array}$$

Interaction laws form a category $\mathbf{IL}(C)$, where the identity on (F,G,ϕ) is $(\mathrm{id}_F, \mathrm{id}_G)$, and the composition of $(f,g) : (F,G,\phi) \to (F',G',\phi')$ and $(f',g') : (F',G',\phi') \to (F'',G'',\phi'')$ is $(f' \circ f, g \circ g')$.

The composition monoidal structure of $[C,C]$ induces a similar monoidal structure on the category $\mathbf{IL}(C)$. The tensorial unit is $(\mathsf{Id}, \mathsf{Id}, \mathrm{id}_{\mathsf{Id}\times\mathsf{Id}})$. The tensor of (F,G,ϕ) and (J,K,ψ) is $(F \cdot J, G \cdot K, \psi \circ \phi \cdot (J \times K))$. The tensor of $(f,g) : (F,G,\phi) \to (F',G',\phi')$ and $(j,k) : (J,K,\psi) \to (J',K',\psi')$ is $(f \cdot j, g \cdot k)$.[2]

2.2 Two degeneracy results

Here are two simple degeneracy results. We first recall the notion of operation for monads and functors.

A comment on operations. The concept of (algebraic) operation of a monad can be defined in several ways. Given a monad T, an *n-ary operation* of T can be defined to be a natural transformation $c'_X : (TX)^n \to TX$ (where X^n is n-fold product of X with itself) satisfying

$$\begin{array}{ccc} (TTX)^n & \xrightarrow{c'_{TX}} & TTX \\ {\scriptstyle (\mu_X)^n}\downarrow & & \downarrow{\scriptstyle \mu_X} \\ (TX)^n & \xrightarrow{c'_X} & TX \end{array}$$

This is the format used by Plotkin and Power [31]. Alternatively, we can say that it is a natural transformation $c_X : X^n \to TX$ and drop the requirement of commutation with μ, as done by Jaskelioff and Moggi [21].

[2] See [22, Sec. 2] for proofs.

In this paper, we prefer to work with operations as maps $c_X : X^n \to TX$ because this format is intuitive and economic in proofs by diagram chasing, but also because it makes sense already for functors and pointed functors. For operations associated to functors, this format allows one to express equations where both sides are applications of an operation to variables, e.g., commutativity. For operations of pointed functors, one can state flat equations, e.g., idempotence of an operation $c_X : X \times X \to FX$ of a pointed functor $F = (F, \eta)$ can be written as $c_X \circ \langle \mathsf{id}_X, \mathsf{id}_X \rangle = \eta_X$.

Functors with a nullary operation. For the functor $\mathsf{Maybe}\, X = (\mathsf{just} : X) + (\mathsf{nothing} : 1)$, it should be clear intuitively that it cannot have a nondegenerate interacting functor: from the element $\mathsf{nothing}_0$ of $\mathsf{Maybe}\, 0$, one cannot possibly extract an element of 0. Formally, the following is a theorem: If a functor F has a nullary operation, i.e., comes with a family of maps $c_X : 1 \to FX$ natural in X,[3] then any interacting functor G is constant zero, i.e., $GY \cong 0$ for any Y. Indeed, for any Y, we have the map

$$GY \xrightarrow{\langle !, \mathsf{id} \rangle} 1 \times GY \xrightarrow{c_0 \times \mathsf{id}} F0 \times GY \xrightarrow{\phi_{0,Y}} 0 \times Y \xrightarrow{\mathsf{fst}} 0$$

Since the initial object of an extensive category is strict (any map to 0 is an isomorphism), we can conclude that $GY \cong 0$. The theorem applies to Maybe since it comes with a nullary operation $\mathsf{nothing}_X : 1 \to \mathsf{Maybe}\, X$.

Functors with a commutative binary operation. A similar no-go theorem holds for commutative binary operations: If a functor F has a commutative binary operation, i.e., comes with a family of maps $c_X : X \times X \to FX$ natural in X such that $c_X = c_X \circ \mathsf{sym}_{X,X}$, then any interacting functor G is constant zero, i.e., $GY \cong 0$ for any Y. Indeed, let $\mathbb{B} = (\mathsf{tt} : 1) + (\mathsf{ff} : 1)$. Then, for any Y, the map

$$\begin{aligned} f_Y \quad = \quad & GY \xrightarrow{\langle !, \mathsf{id} \rangle} 1 \times GY \xrightarrow{\langle \mathsf{tt}, \mathsf{ff} \rangle \times \mathsf{id}} (\mathbb{B} \times \mathbb{B}) \times GY \xrightarrow{c_{\mathbb{B}} \times \mathsf{id}} \\ & F\mathbb{B} \times GY \xrightarrow{\phi_{\mathbb{B},Y}} \mathbb{B} \times Y \xrightarrow{\mathsf{fst}} \mathbb{B} \end{aligned}$$

has the property that $\mathsf{not} \circ f_Y = f_Y$. From this by extensivity, f_Y factors through 0, i.e., we have a map $f'_Y : GY \to 0$, so again $GY \cong 0$.[4]

The degeneracy problem can be overcome by switching to a residual version of interaction laws. Given a monad R on $\mathcal{C}$, an *R-residual functor-functor interaction law* is given by two endofunctors F, G and a family of maps $\phi : FX \times GY \to R(X \times Y)$ natural in X, Y. The monoidal structure of the category $\mathbf{IL}(\mathcal{C}, R)$ of R-residual functor-functor interaction laws relies on the monad structure of R. Typically, one would use the maybe, finite nonempty multiset or finite multiset monad as R.[5]

[3] See [22, Sec. 2] for a more detailed discussion of operations of functors and monads.

[4] For a full proof, see [22, Sec. 2].

[5] For lack of space, we discuss residual functor-functor and monad-comonad interaction only in [22, Sec. 5].

2.3 On the structure of $\mathbf{IL}(\mathcal{C})$

We now look at some ways to construct functor-functor interaction laws systematically.

"Stretching". Given a functor-functor interaction law (F, G, ϕ) and natural transformations $f : F' \to F$ and $g : G' \to G$, we have a functor-functor interaction law $(F', G', \phi \circ (f \times g))$.

Self-duality. For any functor-functor interaction law (F, G, ϕ), we have another functor-functor interaction law $(F, G, \phi)^{\mathrm{rev}} = (G, F, \phi^{\mathrm{rev}})$ where $\phi^{\mathrm{rev}}_{X,Y} = \mathsf{sym}_{Y,X} \circ \phi_{Y,X} \circ \mathsf{sym}_{FX,GY}$. This object mapping extends to maps by $(f, g)^{\mathrm{rev}} = (g, f)$, so we have a functor $(-)^{\mathrm{rev}} : (\mathbf{IL}(\mathcal{C}))^{\mathrm{op}} \to \mathbf{IL}(\mathcal{C})$. The functor $(-)^{\mathrm{rev}}$ is an isomorphism between $(\mathbf{IL}(\mathcal{C}))^{\mathrm{op}}$ and $\mathbf{IL}(\mathcal{C})$.

The final functor-functor interaction law. The final functor-functor interaction law is $(1, 0, \phi)$ where

$$\phi_{X,Y} = \quad 1 \times 0 \xrightarrow{\mathsf{snd}} 0 \xrightarrow{?} X \times Y$$

By self-duality, the initial functor-functor interaction law is $(0, 1, \phi^{\mathrm{rev}})$.

Product of two functor-functor interaction laws. Given two functor-functor interaction laws (F_0, G_0, ϕ_0) and (F_1, G_1, ϕ_1), their product is $(F_0 \times F_1, G_0 + G_1, \phi)$ where

$$\begin{aligned} & \phi_{X,Y} = \\ & (F_0X \times F_1X) \times (G_0Y + G_1Y) \xrightarrow{\mathsf{rdist}} \\ & \quad (F_0X \times F_1X) \times G_0Y + (F_0X \times F_1X) \times G_1Y \xrightarrow{\mathsf{fst} \times \mathsf{id} + \mathsf{snd} \times \mathsf{id}} \\ & \quad F_0X \times G_0Y + F_1X \times G_1Y \xrightarrow{\phi_{0X,Y} + \phi_{1X,Y}} \\ & \quad X \times Y + X \times Y \xrightarrow{\nabla} X \times Y \end{aligned}$$

By self-duality, the *coproduct* of $(G_0, F_0, \phi_0^{\mathrm{rev}})$ and $(G_1, F_1, \phi_1^{\mathrm{rev}})$ is $(G_0 + G_1, F_0 \times F_1, \phi^{\mathrm{rev}})$.

An initial algebra-final coalgebra construction. Assume that $\mathcal{C}$ has the relevant initial algebras and final coalgebras. Given functors $F, G : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ and a family of maps $\phi_{X,Y,W,Z} : F(X, Z) \times G(Y, W) \to X \times Y + Z \times W$ natural in X, Y, Z, W. Then we have an interaction law (F', G', ϕ') where $F'X = \mu Z.\, F(X, Z)$, $G'X = \nu W.\, G(Y, W)$ and ϕ' is constructed as follows. We equip $G'Y \Rightarrow (X \times Y)$ with an $F(X, -)$-algebra structure $\theta^0_{X,Y}$ by currying the map

$$\begin{aligned} & \theta_{X,Y} = \\ & F(X, G'Y \Rightarrow (X \times Y)) \times G'Y \xrightarrow{\mathsf{id} \times \mathsf{out}_{G(Y,-)}} \\ & \quad F(X, G'Y \Rightarrow (X \times Y)) \times G(Y, G'Y) \xrightarrow{\phi_{X,Y,G'Y \Rightarrow (X \times Y), G'Y}} \\ & \quad X \times Y + (G'Y \Rightarrow (X \times Y)) \times G'Y \xrightarrow{\mathsf{id} + \mathsf{ev}} \\ & \quad X \times Y + X \times Y \xrightarrow{\nabla} X + Y \end{aligned}$$

The map $\phi'_{X,Y}$ is obtained by uncurrying the corresponding unique map $\phi^0_{X,Y} : F'X \to G'Y \Rightarrow (X \times Y)$ from the initial $F(X, -)$-algebra.

Restricting to fixed F or G. Sometimes it is of interest to focus on interaction laws of a fixed first functor F or a fixed second functor G (and accordingly on interaction law maps with the first resp. the second natural transformation the identity natural transformation on F resp. G). We denote the corresponding categories by $\mathbf{IL}(C)|_{F,-}$ and $\mathbf{IL}(C)|_{-,G}$. The isomorphism of categories $\mathbf{IL}(C)^{\mathrm{op}} \cong \mathbf{IL}(C)$ given by $(-)^{\mathrm{rev}}$ restricts to $(\mathbf{IL}(C)|_{F,-})^{\mathrm{op}} \cong \mathbf{IL}(C)|_{-,F}$.

The final object of $\mathbf{IL}(C)|_{F,-}$ is $(F, 0, \phi)$ where

$$\phi_{X,Y} = FX \times 0 \xrightarrow{\mathsf{snd}} 0 \xrightarrow{?} X \times Y$$

By self-duality, the initial object of $\mathbf{IL}(C)|_{-,F}$ is $(0, F, \phi^{\mathrm{rev}})$. About the initial object of $\mathbf{IL}(C)|_{F,-}$ we will see in the next subsection.

2.4 Functor-functor interaction in terms of the dual

If C is Cartesian closed, then we can define the *dual* G° of an *endofunctor* G on C by

$$G^\circ X = \int_Y GY \Rightarrow (X \times Y)$$

provided that this end exists, and the *dual* $g^\circ : G^\circ \to G'^\circ$ of a *natural transformation* $g : G' \to G$ by

$$g^\circ_X = \int_Y g_Y \Rightarrow (X \times Y)$$

This construction is contravariantly functorial, i.e., if the dual is everywhere defined, then we have $(-)^\circ : [C, C]^{\mathrm{op}} \to [C, C]$. The existence of all the ends required for this is a strong condition (e.g., a small category that has all limits under classical logic is necessarily a preorder by an argument by Freyd [24]). But for well-definedness and functoriality of $(-)^\circ$ in the general case, it suffices to restrict it to those endofunctors on C that happen to have the dual or, if one so wishes, to some well-delineated smaller class of functors that are guaranteed to have it (e.g., to finitary functors if C is locally finitely presentable). For $(-)^\circ$ to be a contravariant endofunctor on some full subcategory of $[C, C]$, we can restrict it to those endofunctors on C that are dualizable any finite number of times or to some other class of functors closed under the dual. Throughout this paper, we deliberately ignore this existence issue: we either explicitly prove for the ends of interest that they exist or we use such ends on the assumption that they happen to exist.

We have

$$\int_Y C(GY, \overbrace{\textstyle\int_X FX \Rightarrow (Y \times X)}^{F^\circ Y}) \cong \int_{Y,X} C(GY \times FX, Y \times X)$$
$$\cong \int_{X,Y} C(FX \times GY, X \times Y) \cong \int_X C(FX, \underbrace{\textstyle\int_Y GY \Rightarrow (X \times Y)}_{G^\circ X})$$

where by the top-level ends we just indicate collections (not necessarily sets) of natural transformations, so existence is not an issue.

Thus, to have a functor-functor interaction law of F, G is the same as to have a natural transformation $\phi : F \to G^\circ$ or a natural transformation $\phi : G \to F^\circ$.

From the second one of these bijections, we have a canonical interaction between a functor F and its dual F° (take $G = F^\circ$ and $\phi = \mathrm{id}_{F^\circ}$). Moreover, every interaction law between F and G factors uniquely through this canonical interaction law between F and F°.

Under the first of the identifications above, an interaction law map between (F, G, ϕ) and (F', G', ϕ') is given by natural transformations $f : F \to F'$ and $g : G' \to G$ satisfying $g^\circ \circ \phi = \phi' \circ f$. Under the second one, an interaction law map between (F, G, ϕ) and (F', G', ϕ') is given by natural transformations $f : F \to F'$ and $g : G' \to G$ satisfying $\phi \circ g = f^\circ \circ \phi'$.

We have thus established that these categories are isomorphic:

(o) the category $\mathbf{IL}(C)$ of functor-functor interaction laws;
(i) the comma category $[C, C] \downarrow (-)^\circ$ of triples of two functors F, G and a natural transformation $F \to G^\circ$;
(ii) the comma category $(-)^{\circ\mathrm{op}} \downarrow [C, C]^{\mathrm{op}}$ of triples of two functors F, G and a natural transformation $G \to F^\circ$.

From these observations it is immediate that $\mathbf{IL}(C)|_{-,G} \cong [C, C]/G^\circ$ and $\mathbf{IL}(C)|_{F,-} \cong F^\circ\backslash[C, C]^{\mathrm{op}}$. Hence, the initial object of $\mathbf{IL}(C)|_{-,G}$ is $(0, G, \ldots)$ while the final object is $(G^\circ, G, \ldots)$. The initial object of $\mathbf{IL}(C)|_{F,-}$ is $(F, F^\circ, \ldots)$ while the final object is $(F, 0, \ldots)$.

2.5 Dual for some constructions on functors

Here are constructions of the dual for some basic constructions of functors.[6]

Dual of the identity functor. $\mathsf{Id}^\circ \cong \mathsf{Id}$.

Duals of terminal functor, products of a functor, initial functor, coproduct of two functors. Let $G\,Y = 1$. Then $G^\circ X \cong 0$.

Let $G\,Y = A \times G'Y$. Then $G^\circ X \cong A \Rightarrow G'^\circ X$. A little more generally, for $G\,Y = \sum a : A.\, G'a\,Y$, one has $G^\circ X \cong \prod a : A.\, (G'a)^\circ X$.

Specializing to $A = 0$ resp. $A = \mathbb{B}$, we learn: Let $G\,Y = 0$. Then $G^\circ X \cong 1$. Let $G\,Y = G_0\,Y + G_1\,Y$. Then $G^\circ X \cong G_0^\circ X \times G_1^\circ X$.

Dual of exponents of the identity functor. Let $GY = A \Rightarrow Y$. Then $G^\circ X \cong A \times X$.

Example 2.4. Let $G\,Y = Y^+ = \mu W.\, Y \times (1 + W) \cong \sum n : \mathbb{N}.\, ([0..n] \Rightarrow Y)$ (nonempty lists). We have $G^\circ X \cong \prod n : \mathbb{N}.\, ([0..n] \times X)$.

[6] For proofs, see [22, Sec. 2].

Sometimes only a "lower bound" on the dual of a functor constructed from some given functors can be expressed in terms of their duals. This holds for the composition of two general functors, incl. for exponents of a general functor.

Dual of exponents of a general functor. Let $G\,Y = A \Rightarrow G'\,Y$. For a general G', we only have a canonical natural transformation with components $A \times G'^{\circ}\,Y \to G^{\circ}\,Y$.

Dual of composition of two general functors. For general G_0, G_1, we only have the canonical natural transformation $\mathsf{m}^{G_0,G_1} : G_0^{\circ} \cdot G_1^{\circ} \to (G_0 \cdot G_1)^{\circ}$. This hints that $(-)^{\circ} : [C,C]^{\mathrm{op}} \to [C,C]$ is not monoidal, but only lax monoidal (see Section 3.4).

Example 2.5. Let $G_0\,Y = A \Rightarrow Y$, $G_1\,Y = B \times Y$, so $G\,Y = (G_0 \cdot G_1)\,Y = A \Rightarrow (B \times Y) \cong (A \Rightarrow B) \times (A \Rightarrow Y)$. The dual of G is $G^{\circ}\,X \cong (A \Rightarrow B) \Rightarrow (A \times X)$ rather than $(G_0^{\circ} \cdot G_1^{\circ})\,X \cong A \times (B \Rightarrow X)$ as we might perhaps expect. We saw the interaction law of G with G° in Example 2.2. The canonical natural transformation $\mathsf{m}^{G_0,G_1} : G_0^{\circ} \cdot G_1^{\circ} \to G^{\circ}$ is $\mathsf{m}^{G_0,G_1}\,(a,f) = \lambda g.\,(a, f\,(g\,a))$.

3 Monad-comonad interaction

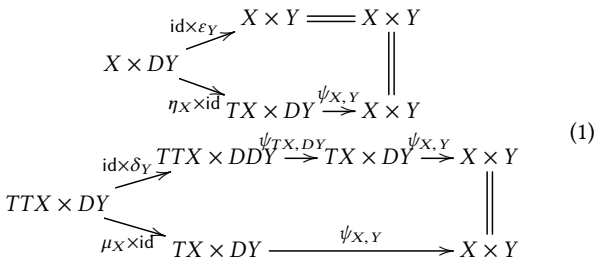
3.1 Monad-comonad interaction laws

In a monad-comonad interaction law, the allowed computations (the chosen notion of computation) must include "just returning" and be closed under sequential composition, so they are defined by a monad rather than a functor. To match this, the allowed machine behaviors (the notion of machine behavior) are defined by a comonad. The idea is that interaction of a "just returning" computation should terminate immediately (in the initial state of the given machine behavior) whereas interaction of a sequence of computations should amount to a sequence of interactions.

We define a *monad-comonad interaction law* on C to be given by a monad $T = (T, \eta, \mu)$ and a comonad $D = (D, \varepsilon, \delta)$ together with a family ψ of maps

$$\psi_{X,Y} : TX \times DY \to X \times Y$$

natural in X and Y (i.e., a functor-functor interaction law of T, D where T and D carry a monad resp. comonad structure) such that also



(1)

Example 3.1. Take $TX = A \Rightarrow X$, $DY = A \times Y$ and $\psi\,(f,(a,y)) = (f\,a, y)$ for a fixed object A. The functors T and D are a monad (a reader monad) resp. a comonad and ψ meets the conditions (1).

Example 3.2. Take $TX = B \times X$, $DY = B \Rightarrow Y$ and $\psi\,((b,x),g) = (x, g\,b)$ for a fixed monoid B. The functors T, D are a monad (a writer monad) resp. a comonad and ψ meets the requisite conditions.

Example 3.3. Take $TX = A \Rightarrow (B \times X)$, $DY = A \times (B \Rightarrow Y)$, $\psi\,(f,(a,g)) = \mathsf{let}\ (b,x) \leftarrow f\,a\ \mathsf{in}\ (x, g\,b)$ for a fixed monoid B acting on a fixed object A. The functors T, D are a monad (an update monad [6]) resp. a comonad and ψ meets the requisite conditions.

Example 3.4. Take $TX = \mu Z.X + (A \Rightarrow (B \times Z))$, $DY = \nu W.Y \times (A \times (B \Rightarrow W))$ for fixed objects A, B. The functors T and D are a free monad and a cofree comonad and the canonical natural transformation ψ satisfies the two equations of a monad-comonad interaction law.

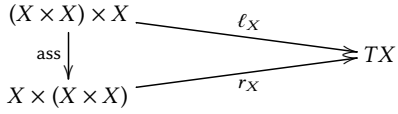
Monad-comonad interaction laws are essentially the same as monoid objects in the monoidal category $\mathbf{IL}(C)$ of functor-functor interaction laws. To be precise, a monad-comonad interaction law $((T,\eta,\mu),(D,\varepsilon,\delta),\psi)$ yields a monoid $((T,D,\psi),(\eta,\varepsilon),(\mu,\delta))$ and vice versa.

A *monad-comonad interaction law map* between (T,D,ψ), (T',D',ψ') is a pair $(f : T \to T', g : D' \to D)$ of a monad map and a comonad map that, as a pair of natural transformations between the underlying functors, is a functor-functor interaction law morphism between the underlying functor-functor interaction laws.

Monad-comonad interaction law maps correspond to monoid morphisms in $\mathbf{IL}(C)$. Thus monad-comonad interaction laws form a category $\mathbf{MCIL}(C)$ isomorphic to the category $\mathbf{Mon}(\mathbf{IL}(C))$.

3.2 A degeneracy result

Monads with an associative operation. Here is a degeneracy theorem for monad-comonad interaction laws:[7] If a monad T has an associative binary operation, i.e., family of maps $c_X : X \times X \to TX$ natural in X satisfying

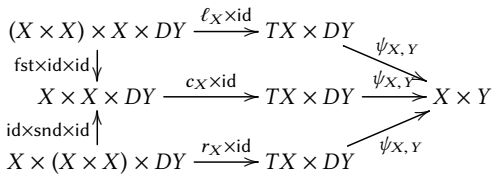


where

$$\ell_X = (X \times X) \times X \xrightarrow{c_X \times \eta_X} TX \times TX \xrightarrow{c_{TX}} TTX \xrightarrow{\mu_X} TX$$
$$r_X = X \times (X \times X) \xrightarrow{\eta_X \times c_X} TX \times TX \xrightarrow{c_{TX}} TTX \xrightarrow{\mu_X} TX$$

[7] See [22, Sec. 3] for a proof.

then, for any comonad D and interaction law $\psi_{X,Y} : TX \times DY \to X \times Y$, we have



Example 3.5. The monad $TX = X^+$ of nonempty lists (the free semigroup delivering monad) comes with an associative operation $\mathrm{dblt}_X : X \times X \to TX$ defined by $\mathrm{dblt}(x_0, x_1) = [x_0, x_1]$. The degeneracy theorem tells us that, while functor-functor interaction laws can accomplish this, no monad-comonad interaction law can extract x_1 from a list $[x_0, x_1, x_2]$ and more generally any middle element x_i $(0 < i < n + 1)$ from a list $[x_0, \ldots, x_{n+1}]$.

Just as functor-functor interaction laws can be generalized to a residual variant to counteract degeneracies, so can monad-comonad interaction laws. For a given monad R, an R-residual interaction law is given by a monad T, a comonad D and a family of maps $\psi_X : TX \times DY \to R(X \times Y)$ natural in X, Y satisfying two equations. To illustrate them at work, let us look at two examples.[8]

Example 3.6. Take $RX = X + 1$ (the maybe monad), $TX = \mu Z.\, X + (1 + Z \times Z)$ (nullary-binary leaf trees for nondeterministic choice), $DY = \nu W.\, Y \times (W + W) \cong (Y \times \mathbb{B})^\omega$ (behaviors of resolvers of binary choice). There is an obvious R-residual monad-comonad interaction law of T and D.

Example 3.7. Take $RX = TX = \nu Z.\, X + Z$ (the delay monad), $DY = \mu W.\, Y \times (1 + W) \cong Y^+$ (nonempty lists with the suffixes comonad structure, the "timeout" comonad). In the R-residual interaction of T and D, a possibly nonterminating computation interacts with a timeout behavior. Should it be interrupted in this process, the outcome is its remainder, with the possible return value in it paired up with the final state (using that R is strong). In this example, suggested to us by Niels Voorneveld, a residual monad is needed because the notion of computations is coinductive.

3.3 On the structure of MCIL(C)

We now explore the structure of the category **MCIL**(C). As this is the category of monoid objects of **IL**(C), the structure of **MCIL**(C) is in many respects similar to **IL**(C). But there are also important differences.

"Stretching". Given a monad-comonad interaction law (T, D, ψ), a monad morphism $f : T' \to T$ and a comonad morphism $g : D' \to D$, we have a monad-comonad interaction law $(T', D', \psi \circ f \times g)$.

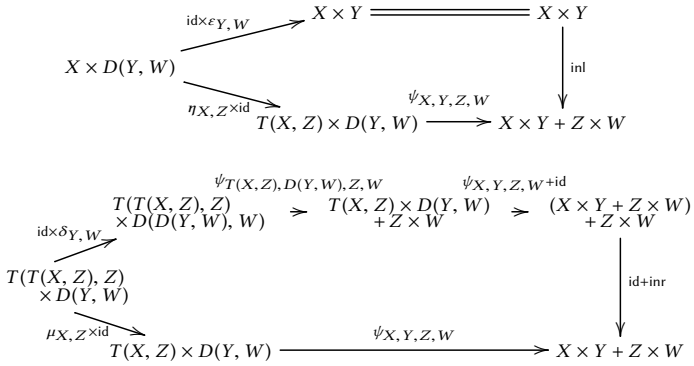
[8] See [22, Sec. 5] for a detailed discussion.

Final and initial monad-comonad interaction laws. The final monad-comonad interaction law is $(1, 0, \psi)$ where $\psi_{X,Y} : 1 \times 0 \to X \times Y$ is the evident map.

The initial monad-comonad interaction law is $(\mathsf{Id}, \mathsf{Id}, \mathsf{id}_{\mathsf{Id} \times \mathsf{Id}})$.

Product of two monad-comonad interact. laws. Given two monad-comonad interaction laws (T_0, D_0, ψ_0) and (T_1, D_1, ψ_1), their product is $(T_0 \times T_1, D_0 + D_1, \psi)$ where $\psi_{X,Y} : (T_0X \times T_1X) \times (D_0Y + D_1Y) \to X \times Y$ is defined as in Section 2. The product of the underlying functors of the two monads is the underlying functor of their product. Likewise, the coproduct of the underlying functors of the two comonads is the underlying functor of their coproduct. The natural transformation ψ agrees with these monad and comonad structures.[9]

An initial algebra-final coalgebra construction. The initial algebra-final coalgebra construction from Sec. 2 gives a monad-comonad interaction law if we start with a parameterized monad T, a parameterized comonad D [40] and a family of maps $\psi_{X,Y,Z,W} : T(X, Z) \times D(Y, W) \to X \times Y + Z \times W$ natural in X, Y, Z, W that agree in the sense of commutation of the diagrams



We get a monad-comonad interaction law (T', D', ψ') where $T'X = \mu Z.\, T(X, Z)$, $D'Y = \nu W.\, D(Y, W)$ and ψ' is defined as in Sec. 2. The functors T' and D' carry monad resp. comonad structures [40] and the natural transformation ψ agrees with those.

Free monad-comonad interaction law. If C has relevant initial algebras and final coalgebras, then, given an interaction law (F, G, ϕ), the free monad-comonad interaction law is provided by the free monad F^* and the cofree comonad $G^\dagger$ and a suitable natural transformation ψ'.

The free monad is given by $F^*X = \mu Z.\, X + FZ$. Its monad structure is induced by the parameterized monad $T(X, Z) = X + FZ$. Similarly, the cofree comonad is given by $G^\dagger Y = \nu W.\, Y \times GW$. Its comonad structure is induced by the parameterized comonad $D(Y, W) = Y \times GW$. In order to construct ψ' following the construction we described in the previous paragraph, we need to construct a family of maps

[9] For the coproduct of two monad-comonad interaction laws and also for monad-comonad interaction laws from distributive laws, see [22, Sec. 3].

$\psi_{X,Y,Z,W} : (X + FZ) \times (Y \times GW) \to X \times Y + Z \times W$ natural in X, Y, Z, W. This is defined as follows:

$$\begin{aligned}&\psi_{X,Y,Z,W} = \\ &(X + FZ) \times (Y \times GW) \xrightarrow{\text{ldist}} \\ &\quad X \times (Y \times GW) + FZ \times (Y \times GW) \xrightarrow{\text{id}\times\text{fst}+\text{id}\times\text{snd}} \\ &\quad X \times Y + FZ \times GW \xrightarrow{\text{id}+\phi_{Z,W}} X \times Y + Z \times W\end{aligned}$$

Restricting to fixed T or D. We denote the categories obtained from $\mathbf{MCIL}(C)$ by fixing the monad T or the comonad D by $\mathbf{MCIL}(C)|_{T,-}$ and $\mathbf{MCIL}(C)|_{-,D}$. The final object of $\mathbf{MCIL}(C)|_{T,-}$ is $(T, 0, \psi)$ where

$$\psi_{X,Y} = TX \times 0 \xrightarrow{\text{snd}} 0 \xrightarrow{?} X \times Y$$

Note that 0 is the initial comonad. The initial object of $\mathbf{MCIL}(C)|_{-,D}$ is (Id, D, ψ) where

$$\psi_{X,Y} = X \times DY \xrightarrow{\text{id}\times\varepsilon_Y} X \times Y$$

This is because Id is the initial monad.

3.4 Monad-comonad interaction in terms of dual and Sweedler dual

Similarly to case of functor-functor interaction laws and maps between them, the dual allows us to obtain useful alternative characterizations of monad-comonad interaction laws and their maps. But a complication arises, see below.[10]

First, let us notice that we have, canonically, a natural transformation $\mathsf{e} : \mathsf{Id} \to \mathsf{Id}^\circ$ and, for any F, G, a natural transformation $\mathsf{m}_{F,G} : F^\circ \cdot G^\circ \to (F \cdot G)^\circ$. These are informally defined by $\mathsf{e}_X x = \lambda_Y.\, \lambda y.\, (x, y) : X \to \int_Y Y \Rightarrow (X \times Y)$ and $(\mathsf{m}_{F,G})_X f = \lambda_{Y'}.\, \lambda z.\, \mathsf{let}\ (g, w) \leftarrow f_{GY}\, z\ \mathsf{in}\ g_Y\, w : \int_{Y'} FY' \Rightarrow (\int_{Y''} GY'' \Rightarrow (X \times Y'')) \times Y' \to \int_Y F(GY) \Rightarrow (X \times Y)$. The natural transformation e is a natural isomorphism; its inverse $\mathsf{e}^{-1} : \mathsf{Id}^\circ \to \mathsf{Id}$ is defined by $\mathsf{e}^{-1}_X f = \mathsf{let}\ (x, \_) \leftarrow f 1 *\ \mathsf{in}\ x : \int_Y Y \Rightarrow (X \times Y) \to X$.

The data (e, m) satisfy the conditions to make $(-)^\circ : [C, C]^{\text{op}} \to [C, C]$ a lax monoidal functor wrt. the $(\mathsf{Id}, \cdot)$ composition monoidal structure of $[C, C]$.

Now, as a first alternative characterization, a monad-comonad interaction law of T and D is essentially the same as a natural transformation $\psi : T \to D^\circ$ satisfying

$$\begin{array}{ccc} \mathsf{Id} & \xrightarrow{\mathsf{e}} & \mathsf{Id}^\circ \\ {\scriptstyle\eta}\downarrow & & \downarrow{\scriptstyle\varepsilon^\circ} \\ T & \xrightarrow{\psi} & D^\circ \end{array} \qquad \begin{array}{ccc} T \cdot T & \xrightarrow{\psi\cdot\psi} & D^\circ \cdot D^\circ \xrightarrow{\mathsf{m}_{D,D}} (D \cdot D)^\circ \\ {\scriptstyle\mu}\downarrow & & \downarrow{\scriptstyle\delta^\circ} \\ T & \xrightarrow{\qquad\psi\qquad} & D^\circ \end{array}$$

Now, since $(-)^\circ : [C, C]^{\text{op}} \to [C, C]$ is lax monoidal, it sends monoids in $[C, C]^{\text{op}}$ to monoids in $[C, C]$, i.e., comonads to monads. In particular, it sends the comonad (D, ε, δ) to the monad $D^\circ = (D^\circ, \varepsilon^\circ \circ \mathsf{e}, \delta^\circ \circ \mathsf{m})$. The conditions above are precisely the conditions for ψ to be a monad map from T to D°. Summing up, a monad-comonad interaction law of T, D amounts to a monad map $\psi : T \to D^\circ$.

As a second alternative, a monad-comonad interaction law of T, D is given by a natural transformation $\psi : D \to T^\circ$ satisfying

$$\begin{array}{ccc} \mathsf{Id} & \xrightarrow{\mathsf{e}} & \mathsf{Id}^\circ \\ {\scriptstyle\varepsilon}\uparrow & & \uparrow{\scriptstyle\eta^\circ} \\ D & \xrightarrow{\psi} & T^\circ \end{array} \qquad \begin{array}{ccc} D \cdot D & \xrightarrow{\psi\cdot\psi} & T^\circ \cdot T^\circ \xrightarrow{\mathsf{m}_{T,T}} (T \cdot T)^\circ \\ {\scriptstyle\delta}\uparrow & & \uparrow{\scriptstyle\mu^\circ} \\ D & \xrightarrow{\qquad\psi\qquad} & T^\circ \end{array} \tag{2}$$

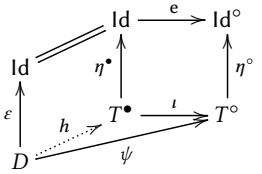
Now, unfortunately, $(-)^\circ$ is not oplax monoidal, so it does not generally send comonoids to comonoids, and T° is generally not a comonad. We could define a candidate counit for T° as $\mathsf{e}^{-1} \circ \eta^\circ : T^\circ \to \mathsf{Id}$, but there is generally no candidate for the comultiplication as we cannot invert $\mathsf{m}_{T,T}$. So we cannot generally say that a monad-comonad interaction law is a comonad map from D to T°; the functor T° is not a comonad.

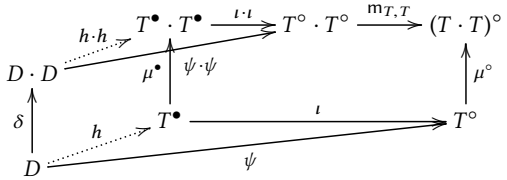
But it may be that there exists what one could informally describe as the greatest comonad smaller (in an appropriate sense) than T°. The formal object of interest here is what we call, following the use of this word in other contexts [18, 34, 35], the Sweedler (or finite) dual of the monad T. It is really just the greatest among all comonads D satisfying conditions (2).

We say that the *Sweedler dual* of the monad T is the (unique up to isomorphism, if it exists) comonad $T^\bullet = (T^\bullet, \eta^\bullet, \mu^\bullet)$ together with a natural transformation $\iota : T^\bullet \to T^\circ$ such that

$$\begin{array}{ccc} \mathsf{Id} & \xrightarrow{\mathsf{e}} & \mathsf{Id}^\circ \\ {\scriptstyle\eta^\bullet}\uparrow & & \uparrow{\scriptstyle\eta^\circ} \\ T^\bullet & \xrightarrow{\iota} & T^\circ \end{array} \qquad \begin{array}{ccc} T^\bullet \cdot T^\bullet & \xrightarrow{\iota\cdot\iota} & T^\circ \cdot T^\circ \xrightarrow{\mathsf{m}_{T,T}} (T \cdot T)^\circ \\ {\scriptstyle\mu^\bullet}\uparrow & & \uparrow{\scriptstyle\mu^\circ} \\ T^\bullet & \xrightarrow{\qquad\iota\qquad} & T^\circ \end{array} \tag{3}$$

and such that, for any comonad $D = (D, \varepsilon, \delta)$ and a natural transformation ψ satisfying conditions (2), there exists a unique comonad map $h : D \to T^\bullet$ satisfying $\psi = \iota \circ h$ as summarized in the following diagrams:





The left-hand diagrams of (3) and (2) are secondary in this definition. In the left-hand diagram of (3), $\eta^\bullet$ is determined by ι as $\eta^\bullet = \mathsf{e}^{-1} \circ \eta^\circ \circ \iota$. The left-hand diagram of (2) commutes trivially when $\psi = \iota \circ h$ for some comonad map h.

Now, if T has the Sweedler dual, there is a bijection between monad-comonad interaction laws of T, D, i.e, natural

[10] We discuss these isomorphisms of categories only on the level of objects here.

transformations $\psi : D \to T^\circ$ satisfying (2), and comonad maps $h : D \to T^\bullet$. Indeed, any natural transformation ψ satisfying (2) induces a unique comonad map h such that $\iota \circ h = \psi$ by definition of $T^\bullet$. On the other hand, for a comonad map h, we get a natural transformation ψ satisfying (2) simply as the composition $\iota \circ h$. These constructions are inverses.

To sum up, we have proved that the following categories are isomorphic:

(o) monad-comonad interaction laws;
(i) triples of a monad T, a comonad D and a monad map from T to D°;
(ii) triples of a monad T, a comonad D and a natural transformation from D to T° subject to conditions (2);
(iii) triples of a monad T, a comonad D and a comonad map from D to $T^\bullet$.

We see that the initial object of $\mathbf{MCIL}(C)|_{-,D}$ is $(\mathsf{Id}, D, \ldots)$ while the final object is $(D^\circ, D, \ldots)$. The initial object of $\mathbf{MCIL}(C)|_{T,-}$ is $(T, T^\bullet, \ldots)$ while the final object is $(T, 0, \ldots)$.

Calculating the Sweedler dual is a complicated matter and we will come to it in Section 5. But here are two examples where the dual of the underlying functor of a monad is not a comonad and the underlying functor of the Sweedler dual differs from the dual.

Example 3.8. In Example 2.4, we saw that the dual of the functor $TX = X^+$ was $T^\circ Y \cong \prod n : \mathbb{N}.\, [0..n] \times Y$. While the functor T is a monad (the free semigroup delivering monad), its dual T° is not a comonad. The Sweedler dual is $T^\bullet Y = Y \times (Y + Y)$, $\eta^\bullet\,(y, \_) = y$, $\mu^\bullet\,(y, \mathsf{inl}\, y') = ((y, \mathsf{inl}\, y'), \mathsf{inl}\,(y', \mathsf{inl}\, y'))$, $\mu^\bullet\,(y, \mathsf{inr}\, y') = ((y, \mathsf{inr}\, y'), \mathsf{inr}\,(y', \mathsf{inr}\, y'))$, with $\iota_Y : T^\bullet Y \to T^\circ Y$ defined by $\iota\,(y, \_)\,0 = (0, y)$, $\iota\,(\_, \mathsf{inl}\, y')\,(n+1) = (0, y')$, $\iota\,(\_, \mathsf{inr}\, y')\,(n+1) = (n+1, y')$. The monad-comonad interaction law $\psi_{X,Y} : TX \times T^\bullet Y \to X \times Y$ is defined by $\psi\,([x_0], (y, \_)) = (x_0, y)$, $\psi\,([x_0, \ldots, x_{n+1}], (\_, \mathsf{inl}\, y')) = (x_0, y')$, $\psi\,([x_0, \ldots, x_{n+1}], (\_, \mathsf{inr}\, y')) = (x_{n+1}, y')$.

Example 3.9. We learned in Example 2.5 that the dual of the functor $TX = A \Rightarrow (B \times X)$ is $T^\circ Y = (A \Rightarrow B) \Rightarrow (A \times Y)$. But the Sweedler dual of T as a monad when B is a monoid acting on A is $T^\bullet Y = A \times (B \Rightarrow Y)$, $\iota\,(a, f) = \lambda g.\, (a, f\,(g\,a))$. In Example 3.3, we showed the monad-comonad interaction law of T and $T^\bullet$.

4 Stateful running

Monad-comonad interaction laws are related to stateful runners as introduced by Uustalu [41]. Next we present the basic facts about runners using the Sweedler dual and then explain the connection to monad-comonad interaction laws.

4.1 Runners

A runner is similar to a monad-comonad interaction law but the allowed machine behaviors are restricted to operate on a fixed state set and their dynamics is also fixed (in the sense that, for any prospective initial state, there is a behavior pre-determined). Only the initial state is not fixed. The state set is manifest but the notion of machine behavior and the pre-determined dynamics are coalesced with the interaction protocol into the natural transformation that is the runner. The runner is a polymorphic function sending any allowed computation and initial state into a return value and a final state.

Given a monad $T = (T, \eta, \mu)$ on C, we call a *(stateful) runner* of T an object Y with a family θ of maps

$$\theta_X : TX \times Y \to X \times Y$$

natural in X, satisfying

$$\begin{array}{ccc} X \times Y & = & X \times Y \\ {\scriptstyle \eta_X \times \mathrm{id}}\downarrow & & \| \\ TX \times Y & \xrightarrow{\theta_X} & X \times Y \end{array} \qquad \begin{array}{ccccc} TTX \times Y & \xrightarrow{\theta_{TX}} & TX \times Y & \xrightarrow{\theta_X} & X \times Y \\ {\scriptstyle \mu_X \times \mathrm{id}}\downarrow & & & & \| \\ TX \times Y & & \xrightarrow{\qquad\theta_X\qquad} & & X \times Y \end{array}$$

Example 4.1. We revisit Ex. 2.5 about the update monad $TX = A \Rightarrow (B \times X)$ defined by an action $\downarrow : A \times B \to A$ of a monoid B on an object A. An *update lens* [5] is an object Y together with maps $lkp : Y \to A$, $upd : Y \times B \to Y$ such that lkp is a map between the B-sets (Y, upd) and $(A, \downarrow)$. Any update lens gives us a runner of T via $\theta_X : (A \Rightarrow (B \times X)) \times Y \to X \times Y$ defined by $\theta\,(f, y) = \mathsf{let}\ (b, x) \leftarrow f\,(lkp\, y)\ \mathsf{in}\ (x, upd\,(y, b))$. In fact, runners of this monad are in a bijection with update lenses and those in turn are essentially the same as coalgebras for the comonad $DY = A \times (B \Rightarrow Y)$.

A *runner map* between (Y, θ), (Y', θ') is a map $f : Y \to Y'$ satisfying

$$\begin{array}{ccc} TX \times Y & \xrightarrow{\theta_X} & X \times Y \\ {\scriptstyle TX \times f}\downarrow & & \downarrow{\scriptstyle X \times f} \\ TX \times Y' & \xrightarrow{\theta'_X} & X \times Y' \end{array}$$

Runners and their maps form a category $\mathbf{Run}(T)$.

Like monad-comonad interaction laws and maps between them, runners and maps between them admit a number of alternative characterizations.[11]

The first one is that runners of T are essentially the same as objects Y endowed with a monad map $\vartheta : T \to \mathrm{St}^Y$ where $\mathrm{St}^Y = (\mathrm{St}^Y, \eta^Y, \mu^Y)$ is the state monad for Y whose underlying functor is defined by $\mathrm{St}^Y X = Y \Rightarrow (X \times Y)$. This is via the bijection of natural transformations

$$\int_X C(TX \times Y, X \times Y) \cong \int_X C(TX, \underbrace{Y \Rightarrow (X \times Y)}_{\mathrm{St}^Y X})$$

Under this bijection, the runner conditions amount to the monad map conditions.

$$\begin{array}{ccc} X & = & X \\ {\scriptstyle \eta_X}\downarrow & & \downarrow{\scriptstyle \eta^Y_X} \\ TX & \xrightarrow{\vartheta_X} & \mathrm{St}^Y X \end{array} \qquad \begin{array}{ccccc} TTX & \xrightarrow{\vartheta_{TX}} & \mathrm{St}^Y TX & \xrightarrow{\mathrm{St}^Y \vartheta_X} & \mathrm{St}^Y \mathrm{St}^Y X \\ {\scriptstyle \mu_X}\downarrow & & & & \downarrow{\scriptstyle \mu^Y_X} \\ TX & & \xrightarrow{\qquad\vartheta_X\qquad} & & \mathrm{St}^Y X \end{array}$$

[11] We discuss them here on the level runners only. In [22, Sec. 4], we develop the isomorphisms of categories in more detail, also accounting for runner maps.

Second, a runner of the monad T is also essentially the same thing as a coalgebra (Y, γ) of the functor T° satisfying the conditions

$$\begin{array}{ccc} Y & \xrightarrow{e_Y} & \mathsf{Id}^\circ Y \\ \| & & \uparrow{\scriptstyle \eta^\circ_Y} \\ Y & \xrightarrow{\gamma} & T^\circ Y \end{array} \qquad \begin{array}{ccccc} Y & \xrightarrow{\gamma} & T^\circ Y & \xrightarrow{T^\circ\gamma} & T^\circ T^\circ Y \xrightarrow{(\mathsf{m}_{T,T})_Y} (T\cdot T)^\circ Y \\ \| & & & & \uparrow{\scriptstyle \mu^\circ_Y} \\ Y & & \xrightarrow{\qquad\qquad\gamma\qquad\qquad} & & T^\circ Y \end{array} \tag{4}$$

This is because of the bijection

$$\begin{aligned} \textstyle\int_X C(TX\times Y, X\times Y) &\cong \textstyle\int_X C(Y\times TX, Y\times X) \\ &\cong C(Y, \underbrace{\textstyle\int_X TX \Rightarrow (Y\times X)}_{T^\circ Y}) \end{aligned}$$

Recall that the functor T° is generally not a comonad as $\mathsf{m}_{T,T}$ is not invertible, so we cannot generally speak of functor coalgebras satisfying conditions (4) as comonad coalgebras.

Lastly, recall that the costate comonad for an object Y is defined by $\mathrm{Cost}^Y = (\mathrm{Cost}^Y, \varepsilon^Y, \delta^Y)$ is defined by $\mathrm{Cost}^Y Z = (Y \Rightarrow Z)\times Y$, $\varepsilon^Y(f, y) = f\, y$, $\delta^Y(f, y) = (\lambda y'.\,(f, y'), y)$. This gives us a third characterization: a runner is essentially the same as an object Y together with a natural transformation ζ between the underlying functor of the costate comonad Cost^Y and the functor T° satisfying

$$\begin{array}{ccc} \mathsf{Id} & \xrightarrow{e} & \mathsf{Id}^\circ \\ {\scriptstyle\varepsilon^Y}\uparrow & & \uparrow{\scriptstyle\eta^\circ} \\ \mathrm{Cost}^Y & \xrightarrow{\zeta} & T^\circ \end{array} \qquad \begin{array}{ccccc} \mathrm{Cost}^Y\cdot\mathrm{Cost}^Y & \xrightarrow{\zeta\cdot\zeta} & T^\circ\cdot T^\circ & \xrightarrow{\mathsf{m}_{T,T}} & (T\cdot T)^\circ \\ {\scriptstyle\delta^Y}\uparrow & & & & \uparrow{\scriptstyle\mu^\circ} \\ \mathrm{Cost}^Y & & \xrightarrow{\qquad\zeta\qquad} & & T^\circ \end{array} \tag{5}$$

This is because of the bijection

$$\begin{aligned} C(Y, T^\circ Y) &\cong \textstyle\int_Z C(Y\Rightarrow Z, Y\Rightarrow T^\circ Z) \\ &\cong \textstyle\int_Z C(\underbrace{(Y\Rightarrow Z)\times Y}_{\mathrm{Cost}^Y Z}, T^\circ Z) \end{aligned}$$

(The first bijection is an internal version of the Yoneda lemma, which applies as T° is necessarily strong.) If the Sweedler dual comonad $T^\bullet$ of the monad T exists, then we can continue this reasoning. We see that a runner is the essentially the same as an object Y with a comonad morphism between Cost^Y and $T^\bullet$ and that is further essentially the same as an object Y with a comonad coalgebra of $T^\bullet$.

Summing up, we have established that the following categories are isomorphic:

(o) runners of T;
(i) objects Y with a monad map from T to St^Y;
(ii) functor coalgebras of T° subject to conditions (4);
(iii) objects Y with a natural transformation from Cost^Y to T° subject to conditions (5);
(iv) objects Y with a comonad map from Cost^Y to $T^\bullet$;
(v) comonad coalgebras of $T^\bullet$.

4.2 Runners vs. monad-comonad interaction laws

Monad-comonad interaction laws of T, D are in a bijection with *D-coalgebraic T-runner specs* by which we mean carrier-preserving functors between $\mathbf{Coalg}(D)$ and $\mathbf{Run}(T)$, i.e., functors $\Psi : \mathbf{Coalg}(D) \to \mathbf{Run}(T)$ such that

$$\begin{array}{ccc} \mathbf{Coalg}(D) & \xrightarrow{\Psi} & \mathbf{Run}(T) \\ & {\scriptstyle U}\searrow \quad \swarrow{\scriptstyle U} & \\ & C & \end{array}$$

Indeed, given a monad-comonad interaction law ψ, we can define a runner spec Ψ by

$$(\Psi(Y,\gamma))_X = (Y,\ TX\times Y \xrightarrow{\mathsf{id}\times\gamma} TX\times DY \xrightarrow{\psi_{X,Y}} X\times Y\ )$$

In the opposite direction, given a runner spec Ψ, we build a interaction law from the cofree coalgebras of D. For any Y, we have the cofree coalgebra (DY, δ_Y) and define a monad-comonad interaction law ϕ by

$$\phi_{X,Y} = \ TX\times DY \xrightarrow{\Psi(DY,\delta_Y)_X} X\times DY \xrightarrow{\mathsf{id}\times\varepsilon_Y} X\times Y$$

A pair of a monad map $f : T \to T'$ and a comonad map $g : D' \to D$ is an interaction law map between (T, D, ψ) and (T', D', ψ') iff the corresponding coalgebraic runner specs satisfy

$$\begin{array}{ccc} \mathbf{Coalg}(D) & \xrightarrow{\Psi} & \mathbf{Run}(T) \\ {\scriptstyle\mathbf{Coalg}(g)}\uparrow & & \uparrow{\scriptstyle\mathbf{Run}(f)} \\ \mathbf{Coalg}(D') & \xrightarrow{\Psi'} & \mathbf{Run}(T') \end{array}$$

(Notice that $\mathbf{Coalg}(-) : \mathbf{Comnd}(C) \to \mathbf{CAT}$ and $\mathbf{Run}(-) : (\mathbf{Mnd}(C))^{\mathrm{op}} \to \mathbf{CAT}$.) So the categories of monad-comonad interaction laws and coalgebraic runner specs are isomorphic.

More modularly, but assuming that all Sweedler duals exist, the isomorphism of the categories of monad-comonad interaction laws and coalgebraic runner specs follows from the following sequence of isomorphisms of categories, using that $\mathbf{Run}(T) \cong \mathbf{Coalg}(T^\bullet)$:

(o) monad-comonad interaction laws;
(i) triples of a monad T, a comonad D and a comonad map between D, $T^\bullet$;
(ii) triples of a monad T, a comonad D and a carrier-preserving functor between $\mathbf{Coalg}(D)$, $\mathbf{Coalg}(T^\bullet)$;
(iii) coalgebraic runner specs.

5 Monoid-comonoid interaction

Exploiting that monads and monad-like objects like arrows or lax monoidal functors ("applicative functors") are monoids has turned out to be very fruitful in categorical semantics (see, e.g., [11, 21, 37]). We now explore this perspective by abstracting monad-comonad interaction laws into monoid-comonoid interaction laws. This leads us to further known concepts and methods from category theory.

5.1 Interaction laws and Chu spaces

The first step in generalizing interaction laws to monoids and comonoids is to account for interaction laws as maps in a general category. Recall that the Day convolution [13] of functors $F, G : C \to C$ where C is a category with finite products is given by

$$(F \star G)Z = \int^{X,Y} C(X \times Y, Z) \bullet (FX \times GY)$$

provided that this coend exists. If C is Cartesian closed and has enough coends, then $([C,C], \mathsf{Id}, \star)$ is a symmetric monoidal category. Moreover, with enough ends, the functor $- \star G$ has as a right adjoint the functor $G \mathbin{\star\!\!\rightarrow} -$ defined by

$$(G \mathbin{\star\!\!\rightarrow} H)X = \int_Y GY \Rightarrow H(X \times Y)$$

Note that $G^\circ = G \mathbin{\star\!\!\rightarrow} \mathsf{Id}$. (Like in Section 2, if C does not have all necessary coends and ends, one has to restrict $[C,C]$, e.g., to the full subcategory of finitary functors if C lfp, cf. [15].)

By reasoning about natural transformations, we see that interaction laws for a pair of functors F and G amount to maps $\phi : F \star G \to \mathsf{Id}_C$:

$$\begin{aligned} &\int_{X,Y} C(FX \times GY, X \times Y) \\ &\quad\cong \int_{X,Y,Z} \mathbf{Set}(C(X \times Y, Z), C(FX \times GY, Z)) \\ &\quad\cong \int_Z C((F \star G)Z, Z) \end{aligned}$$

We see that a functor-functor interaction law is a triple $(F, G, \phi : F \star G \to \mathsf{Id}_C)$, i.e., a Chu space [7] over the object Id_C wrt. the Day convolution symmetric monoidal closed structure on $[C,C]$. An interaction law map is a Chu space map under this view, so the category $\mathbf{IL}(C)$ is isomorphic to the category $\mathbf{Chu}([C,C], \mathsf{Id}_C)$.

This is nice, but not fine-grained enough for developing an abstract foundation for our theory. The canonical symmetric monoidal structure on $\mathbf{Chu}(\mathcal{F}, H)$ is based on the symmetric monoidal closed structure of the base category $\mathcal{F}$, which in our case is the Day convolution, and uses pullbacks. But we are interested in a different monoidal structure on $\mathbf{IL}(C)$, the one based on composition that has monad-comonad interaction laws as monoids. We introduce this monoidal structure through the characterization of Chu categories as comma categories by Pavlovic [29] and Hasegawa's result about monoidal structures on categorical glueing [17].

5.2 Interaction laws and Hasegawa's glueing

Pavlovic [29] noticed that, for any monoidal closed category $(\mathcal{F}, \star, \mathbin{\star\!\!\rightarrow})$ and an object R of $\mathcal{F}$, $\mathbf{Chu}(\mathcal{F}, R)$ is isomorphic to the comma category $\mathcal{F} \downarrow (-)^\circ$ where $G^\circ = G \mathbin{\star\!\!\rightarrow} R$. We next assume another monoidal structure $(I, \otimes)$ on $\mathcal{F}$, and suitable data to make $(-)^\circ$ a lax monoidal functor of type $(\mathcal{F}, I, \otimes)^{\mathrm{op}} \to (\mathcal{F}, I, \otimes)$. The following construction by Hasegawa [17] (cf. also [3]) allows us to equip $\mathcal{F} \downarrow (-)^\circ$ with a monoidal structure derived from $(I, \otimes)$.

Given two monoidal categories $\mathcal{F} = (\mathcal{F}, I^{\mathcal{F}}, \otimes^{\mathcal{F}})$, $\mathcal{G} = (\mathcal{G}, I^{\mathcal{G}}, \otimes^{\mathcal{G}})$ and a lax monoidal functor $((-)^\circ, \mathsf{e}, \mathsf{m}) : \mathcal{G} \to \mathcal{F}$. The comma category $\mathcal{F} \downarrow (-)^\circ$ carries a monoidal structure given by

$$\begin{aligned} I &= (I^{\mathcal{F}}, I^{\mathcal{G}},\ I^{\mathcal{F}} \xrightarrow{\mathsf{e}} (I^{\mathcal{G}})^\circ\ ) \\ (F, G, \phi) \otimes (F', G', \phi') &= (F \otimes^{\mathcal{F}} F', G \otimes^{\mathcal{G}} G', \\ &\qquad F \otimes^{\mathcal{F}} F' \xrightarrow{\phi \otimes^{\mathcal{F}} \phi'} G^\circ \otimes^{\mathcal{F}} G'^\circ \xrightarrow{\mathsf{m}_{G,G'}} (G \otimes^{\mathcal{G}} G')^\circ\ ) \end{aligned}$$

We carry out this construction of a monoidal structure on the comma category, starting from the duoidal category $(\mathcal{F}, I, \otimes, J, \star)$ closed wrt. $\star$ [3, 15]. This is a category with two monoidal structures, and among its data are a map $\chi : I \star I \to I$ and a family of maps $\xi_{F,F',G,G'} : (F \otimes F') \star (G \otimes G') \to (F \star G) \otimes (F' \star G')$ natural in F, F', G, G'. If now R is an object of $\mathcal{F}$ with a monoid structure (η^R, μ^R) wrt. $(I, \otimes)$, the functor $(-)^\circ : \mathcal{F}^{\mathrm{op}} \to \mathcal{F}$ given by $G^\circ = G \mathbin{\star\!\!\rightarrow} R$ is lax monoidal wrt. the $(I, \otimes)$ monoidal structure since as witnesses e, m of lax monoidality we have the curryings of

$$\begin{aligned} \mathsf{e}' &= I \star I \xrightarrow{\chi} I \xrightarrow{\eta^R} R \\ \mathsf{m}'_{G,G'} &= (G^\circ \otimes G'^\circ) \star (G \otimes G') \xrightarrow{\xi} \\ &\qquad (G^\circ \star G) \otimes (G'^\circ \star G') \xrightarrow{\mathsf{ev} \otimes \mathsf{ev}} R \otimes R \xrightarrow{\mu^R} R \end{aligned}$$

As $\mathcal{F}, \mathcal{G} = \mathcal{F}^{\mathrm{op}}$ and $(-)^\circ$ in the presence of a monoid structure wrt. $(I, \otimes)$ on R fulfill the assumptions of the construction, $\mathcal{F} \downarrow (-)^\circ$ acquires the desired $(I, \otimes)$ based monoidal structure.

To recover R-residual monad-comonad interaction laws, we take $(\mathcal{F}, I, \otimes, J, \star, \mathbin{\star\!\!\rightarrow})$ to be $[C,C]$ with its composition monoidal and Day convolution symmetric monoidal closed structures and R to be a monad. Then from the isomorphism

$$\mathcal{F} \downarrow (-)^\circ \cong \mathbf{Chu}([C,C], R) \cong \mathbf{IL}_R(C),$$

the Chu category acquires a composition-based monoidal structure. A monoid with respect to this monoidal structure is what we will recognize as a R-residual monoid-comonoid interaction law in the duoidal category $\mathcal{F}$.

If $R = \mathsf{Id}$, the notions of dual and Sweedler dual emerge as follows. As the $\star$ monoidal structure is symmetric, we have

$$\mathcal{F}^{\mathrm{op}} \underset{(-)^{\circ \mathrm{op}}}{\overset{(-)^\circ}{\rightleftarrows}} \mathcal{F} \quad (\top)$$

since, for any $F, G \in |[C,C]|$,

$$\begin{aligned} \mathcal{F}(F, G^\circ) &\cong \mathcal{F}(F \star G, \mathsf{Id}) \\ &\cong \mathcal{F}(G \star F, \mathsf{Id}) \cong \mathcal{F}(G, F^\circ) \cong \mathcal{F}^{\mathrm{op}}(F^\circ, G) \end{aligned}$$

Because of this adjunction, we call $(-)^\circ$ the *dual*.

Since the functor $(-)^\circ$ is lax monoidal, it lifts to a functor between the respective categories of monoids (bear in mind that $\mathbf{Mon}(\mathcal{F}^{\mathrm{op}}) = (\mathbf{Comon}(\mathcal{F}))^{\mathrm{op}}$):

$$\begin{array}{ccc} (\mathbf{Comon}(\mathcal{F}))^{\mathrm{op}} & \xrightarrow{(-)^\circ} & \mathbf{Mon}(\mathcal{F}) \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathcal{F}^{\mathrm{op}} & \xrightarrow{(-)^\circ} & \mathcal{F} \end{array} \tag{6}$$

However, its left adjoint $(-)^{\circ\mathrm{op}}$ is only oplax monoidal, but not lax monoidal, so we cannot get a similar diagram for $(-)^{\circ\mathrm{op}}$. We want to find a substitute for this lifting, in particular, we want a left adjoint for the lifted $(-)^{\circ}$:

$$(\mathbf{Comon}(\mathcal{F}))^{\mathrm{op}} \underset{(-)^{\bullet\mathrm{op}}}{\overset{(-)^{\circ}}{\top}} \mathbf{Mon}(\mathcal{F})$$

We obtain not a natural isomorphism between two functors $(\mathbf{Comon}(\mathcal{F}))^{\mathrm{op}} \to \mathcal{F}$ as in diagram (6), but instead only a natural transformation $\iota : (-)^{\circ\mathrm{op}} \cdot U \to U \cdot (-)^{\bullet\mathrm{op}}$ between two functors $\mathbf{Mon}(\mathcal{F}) \to \mathcal{F}^{\mathrm{op}}$.

$$\begin{array}{ccc} (\mathbf{Comon}(\mathcal{F}))^{\mathrm{op}} & \xleftarrow{(-)^{\bullet\mathrm{op}}} & \mathbf{Mon}(\mathcal{F}) \\ {\scriptstyle U}\downarrow & \Downarrow \iota & \downarrow{\scriptstyle U} \\ \mathcal{F}^{\mathrm{op}} & \xleftarrow[(-)^{\circ\mathrm{op}}]{} & \mathcal{F} \end{array}$$

We call the functor $(-)^{\bullet} : (\mathbf{Mon}(F))^{\mathrm{op}} \to \mathbf{Comon}(F)$ the *Sweedler dual*.

5.3 Sweedler dual for some constructions of monoids

As we have seen in the setting of monad-comonad interaction laws, it is not always easy to find the Sweedler dual. In the remainder of this section, we focus on the cases of free monoids and free monoids quotiented by "equations" for one method to compute them.

Let F^* be the free monoid on F. In this case, if the cofree comonoid on F° exists, then it is the Sweedler dual of F^*, i.e., we can show that $(F^*)^\bullet = (F^\circ)^\dagger$. This is seen from the following calculation:

$$\begin{aligned} &(\mathbf{Comon}(\mathcal{F}))^{\mathrm{op}}((F^\circ)^\dagger, D) \\ &\quad\cong\ \mathbf{Comon}(\mathcal{F})(D, (F^\circ)^\dagger) \ \cong\ \mathcal{F}(UD, F^\circ) \\ &\quad\cong\ \mathcal{F}^{\mathrm{op}}(F^\circ, UD) \ \cong\ \mathcal{F}(F, (UD)^\circ) \\ &\quad\cong\ \mathcal{F}(F, UD^\circ) \ \cong\ \mathbf{Mon}(\mathcal{F})(F^*, D^\circ) \end{aligned}$$

This observation facilitates calculation of Sweedler duals of free monads (i.e., theories without equations). A natural question is to ask what happens in the presence of equations. Suppose that we have a monoid T given as a coequalizer

$$E^* \underset{g^L}{\overset{f^L}{\rightrightarrows}} F^* \longrightarrow T$$

in $\mathbf{Mon}(\mathcal{F})$ where $(-)^L$ is the left transpose of the free / forgetful adjunction between $\mathcal{F}$ and $\mathbf{Mon}(\mathcal{F})$. The maps $f, g : E \to UF^*$ of $\mathcal{F}$ represent a system of equations in a set of variables E, and we can think of T as being the monoid obtained by calculating the free monoid and then quotienting by the equations. We can try to obtain the Sweedler dual of T by constructing a "dual" diagram as follows. We can instantiate ι at F^* and obtain a map $\iota_{F^*} : (UF^*)^\circ \to U((F^*)^\bullet)$ in $\mathcal{F}^{\mathrm{op}}$, i.e., a map $\iota_{F^*} : U((F^*)^\bullet) \to (UF^*)^\circ$ in $\mathcal{F}$. By composing with f° and g°, we get:

$$U((F^\circ)^\dagger) = U((F^*)^\bullet) \xrightarrow{\iota_{F^*}} (UF^*)^\circ \underset{g^\circ}{\overset{f^\circ}{\rightrightarrows}} E^\circ$$

The Sweedler dual $T^\bullet$ of T is now obtained as an equalizer in $\mathbf{Comon}(\mathcal{F})$ by

$$T^\bullet \longrightarrow (F^\circ)^\dagger \underset{(g^\circ \circ \iota_{F^*})^R}{\overset{(f^\circ \circ \iota_{F^*})^R}{\rightrightarrows}} (E^\circ)^\dagger$$

where $(-)^R$ is the right transpose of the forgetful/cofree adjunction between $\mathbf{Comon}(\mathcal{F})$ and $\mathcal{F}$.

Example 5.1. Revisiting Example 3.8, the nonempty list monad $TX = X^+$ arises as the quotient of the free monad $T_0X = \mu Z.\, X + Z \times Z$ by the associativity equation for its operation $c_X : X \times X \to T_0X$. The Sweedler dual of T is $T^\bullet Y = Y \times (Y + Y)$. It is the subcomonad of the cofree comonad $T_0^\bullet Y = \nu W.\, Y \times (W + W)$ by the coassociativity coequation for its cooperation $c'_Y : T_0^\bullet Y \to Y + Y$. The comonad $T^\bullet$ is relatively degenerate because coassociativity entails corectangularity (while associativity does not entail rectangularity).[12]

Example 5.2. Going back to Example 3.9, the update monad $TX = A \Rightarrow (B \times X)$ where B is a monoid acting on A arises as the quotient of the free monad $T_0X = \mu Z.\, X + (A \Rightarrow Z) + (B \times Z)$ by three equations for its operations $c_X : (A \Rightarrow X) \to T_0X$ and $d_X : A \times X \to T_0X$. The Sweedler dual of the monad T is $T^\bullet Y = A \times (B \Rightarrow Y)$. It is the subcomonad of the cofree comonad $T_0^\bullet Y = \nu W.\, Y \times (A \times W) \times (B \Rightarrow W)$ by three coequations for its cooperations $c'_Y : T^\bullet Y \to A \times Y$ and $d'_Y : T^\bullet Y \to B \Rightarrow Y$.[13]

6 Related work

Works closest related to this paper on monad-comonad interaction laws are Power and Shkaravska's work on lenses as comodels [36], Power and Plotkin's study of tensors of models and comodels [32], Abou-Saleh and Pattinson's [1] work on comodels for operational semantics, Møgelberg and Staton's work on linear usage of state [25] and Uustalu's work on runners [41]—the starting point for this work.

Power and Shkaravska [36] observed that lenses (which they called arrays) are the same as comodels of the Lawvere theory of state (coalgebras of costate comonads). Hyland, Plotkin and Power [19] had earlier studied tensors of models of Lawvere theories. Given the result about lenses, they felt that a mixed tensor could be relevant for operational semantics and proposed a definition. In the examples, they only considered tensors with free models, which is the relevant case. Abou-Saleh and Pattinson [1] employed comodels and the

[12] In band theory, left and right rectangularity are the equations $(x * y) * z = x * z$ and $x * (y * z) = x * z$.

[13] See [22, Sec. 6] for more detail on these examples.

tensor of free models with comodels for operational semantics. Møgelberg and Staton [25] described a fully-complete state-passing translation from a call-by-value language with generic effects to an enriched call-by-value language with linearly used state based on comodels of the Lawvere theory. Uustalu [41] defined runners as a concept of independent interest and showed that runners of a finitary monad are in bijection with comodels of the corresponding Lawvere theory (coalgebras of the comonad they determine), which was also observed in [25]. He considered a number of concrete theories (the semigroup theory, the band theory etc.) and worked out the corresponding comonads. Pattinson and Schröder [28] studied equational reasoning about coterms and gave a generic complete axiomatization of bisimilarity.

In this paper, we chose to avoid discussing monads and comonads in terms of models and comodels of Lawvere theories. But our construction in Section 5 of the Sweedler dual of the quotient of a free monoid by "equations" as an equalizer entails it as a special case that the Sweedler dual of a finitary monad is the comonad given by the comodels of the corresponding Lawvere theory. It was this decision to work on the abstraction level of monads and comonads (and higher) that made it possible for us to recognize monad-comonad interaction laws as monoid objects in a category arising from Hasegawa's construction.

Runners and therefore also interaction laws are similar to Plotkin and Pretnar's effect handlers [33] in that both are about dealing with effects, but otherwise very different. The big conceptual difference, which also explains why they have to be different technically, is that handlers are for dealing with effects inside a computation while runners and interaction laws are about having effects performed externally.[14] Bauer in his excellent tutorial [8] has made this point very strongly [8]. Effect handlers were first implemented in a programming language Eff by Bauer and Pretnar [9] and there is now a flurry of research in this area. Ahman and Bauer in their new work [4] proposed a language design for (residual) runners; they also implemented a prototype programming language Coop.

Canonical interaction between duals or general interaction have been considered many times in different contexts on different levels of explicitness. Oles [27], who invented lenses (he called them store shapes), did so in order to describe the semantics of Algol. Hancock and Hyvernat's work [16] on interaction structures centers on the canonical interaction law of the free monad on F° and the cofree comonad on F where F is a container functor. (Intuitionistic) linear-logic based two-party session typing [39, 42] is very much about canonical interaction between syntactically dual functors.[15] The same idea is central in game-theoretic semantics of (intuitionistic) linear logic (formulae-as-games, proofs-as-strategies) [2, 14]; glueing and related constructions are a major tool, cf. [20].

The dual of a vector space is a classical construction in algebra. The restriction to the finite (or Sweedler) dual was studied originally by Sweedler in the theory of Hopf algebras [38], as well as the notion of measuring morphism that is analogous to interaction laws. Recently, Porst et al. [35] and Hyland et al. [18] have worked out generalizations of these constructions in categorical terms.

7 Conclusion and future work

We hope to have demonstrated that monad-comonad interaction laws are a natural concept for describing interaction of effectful computations with machines providing the effects. They are well-motivated not only as a computational model, but also mathematically, admitting an elegant theory based on concepts and methods that have previously proved useful in other mathematical contexts, such as the Sweedler dual.

There are many questions that we have not yet answered. What are some general ways to compute the Sweedler dual? Power's work [36] suggests a sophisticated iterative construction based on improving approximations. What is a good general syntax for cooperations and coequations? What can be said about the "dual" and the Sweedler "dual" in the presence of a residual monad and how to compute them in this situation? How to compute the Sweedler dual in some (intuitionistic) linear setting adequate for session typing? A further set of questions concerns handlers. We know how interaction laws and handlers differ, but what are some relationships? For example, what could be some general principles for turning interaction laws into handlers?

Acknowledgments

We are grateful to Robin Cockett for discussions and encouragement and to Ignacio López Franco for pointing out the categorical work on measuring morphisms.

S.K. was supported by the Japan Science and Technology agency ERATO project no. JPMJER1603 (HASUO Metamathematics for Systems Design Project). T.U. was supported by the Icelandic Research Fund project grant no. 196323-051, the Estonian Ministry of Education and Research institutional research grant no. IUT33-13, a project of the Estonian-French Parrot cooperation programme and a guest professorship from Université Paris 13. E.R. was in part supported by the European Research Council starting grant no. 715753 (SECOMP) and by Nomadic Labs via a grant on "Evolution, Semantics, and Engineering of the F^* Verification System". E.R. also benefited from the above-mentioned Parrot and Icelandic Research Fund projects.

[14] See [22, Sec. 7] for a technical discussion of this.
[15] See [22, Sec. 7] for some discussion of this.

References

[1] Faris Abou-Saleh and Dirk Pattinson. 2013. Comodels and Effects in Mathematical Operational Semantics. In *Proc. of 16th Int. Conf. on Foundations of Software Science and Computation Structures, FoSSaCS 2013*, Frank Pfenning (Ed.). Lect. Notes in Comput. Sci., Vol. 7794. Springer, Berlin, Heidelberg, 129–144. https://doi.org/10.1007/978-3-642-37075-5_9

[2] Samson Abramsky and Radha Jagadeesan. 1994. Games and Full Completeness for Multiplicative Linear Logic. *J. Symb. Log.* 59, 2 (1994), 543–574. https://doi.org/10.2307/2275407

[3] Marcelo Aguiar and Swapneel Mahajan. 2010. *Monoidal Functors, Species and Hopf Algebras.* CRM Monograph Series, Vol. 29. Amer. Math. Soc., Providence, RI.

[4] Danel Ahman and Andrej Bauer. 2020. Runners in Action. In *Proc. of 29th European Symp. on Programming, ESOP 2020*, Peter Müller (Ed.). Lect. Notes in Comput. Sci., Vol. 12075. Springer, Cham, 29–55. https://doi.org/10.1007/978-3-030-44914-8_2

[5] Danel Ahman and Tarmo Uustalu. 2014. Coalgebraic Update Lenses. *Electron. Notes Theor. Comput. Sci.* 308 (2014), 25–48. https://doi.org/10.1016/j.entcs.2014.10.003

[6] Danel Ahman and Tarmo Uustalu. 2014. Update Monads: Cointerpreting Directed Containers. In *Proc. of 19th Int. Wksh. on Types for Proofs and Programs, TYPES 2013*, Ralph Matthes and Aleksy Schubert (Eds.). Leibniz Int. Proc. in Informatics, Vol. 26. Dagstuhl Publishing, Saarbrücken/Wadern, 1–23. https://doi.org/10.4230/lipics.types.2013.1

[7] Michael Barr. 2006. The Chu Construction: History of an Idea. *Theory Appl. Categ.* 17, 1 (2006), 10–16. http://www.tac.mta.ca/tac/volumes/17/1/17-01abs.html

[8] Andrej Bauer. 2018. What Is Algebraic about Algebraic Effects and Handlers? arXiv preprint 1807.05923. https://arxiv.org/abs/1807.05923

[9] Andrej Bauer and Matija Pretnar. 2015. Programming with Algebraic Effects and Handlers. *J. Log. Algebr. Methods Program.* 84, 1 (2015), 108–123. https://doi.org/10.1016/j.jlamp.2014.02.001

[10] Mario Cáccamo and Glynn Winskel. 2001. A Higher-Order Calculus for Categories. In *Proc. of 14th Int. Conf. on Theorem Proving in Higher Order Logics, TPHOLs 2001*, Richard J. Boulton and Paul B. Jackson (Eds.). Lect. Notes in Comput. Sci., Vol. 2152. Springer, Berlin, Heidelberg, 136–153. https://doi.org/10.1007/3-540-44755-5_11

[11] Paolo Capriotti and Ambrus Kaposi. 2014. Free Applicative Functors. In *Proc. of 5th Workshop on Mathematically Structured Functional Programming, MSFP 2014*, Paul B. Levy and Neel Krishnaswami (Eds.). Electronic Proc. in Theoretical Computer Science, Vol. 153. Open Publishing Association, Sydney, 2–30. https://doi.org/10.4204/eptcs.153.2

[12] Aurelio Carboni, Stephen Lack, and Robert F.C. Walters. 1993. Introduction to Extensive and Distributive Categories. *J. Pure Appl. Alg.* 84, 2 (1993), 145–158. https://doi.org/10.1016/0022-4049(93)90035-r

[13] Brian Day. 1970. On Closed Categories of Functors. In *Reports of the Midwest Category Seminar IV*, Saunders Mac Lane (Ed.). Lecture Notes in Mathematics, Vol. 137. Springer, Berlin, Heidelberg, 1–38. https://doi.org/10.1007/bfb0060438

[14] Valeria C. V. de Paiva. 1991. *The Dialectica Categories.* Technical Report UCAM-CL-TR-213. Computer Laboratory, University of Cambridge. https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-213.pdf

[15] Richard Garner and Ignacio López Franco. 2016. Commutativity. *J. Pure Appl. Alg.* 220, 5 (2016), 1707–1751. https://doi.org/10.1016/j.jpaa.2015.09.003

[16] Peter Hancock and Pierre Hyvernat. 2006. Programming Interfaces and Basic Topology. *Ann. Pure Appl. Logic* 137, 1–3 (2006), 189–239. https://doi.org/10.1016/j.apal.2005.05.022

[17] Masahito Hasegawa. 1999. Logical Predicates for Intuitionistic Linear Type Theories. In *Proc. of 4th Int. Conf. on Typed Lambda Calculi and Applications, TLCA '99*, Jean-Yves Girard (Ed.). Lect. Notes in Comput. Sci., Vol. 1581. Springer, Berlin, Heidelberg, 198–213. https://doi.org/10.1007/3-540-48959-2_15

[18] Martin Hyland, Ignacio López Franco, and Christina Vasilakopoulou. 2017. Hopf Measuring Comonoids and Enrichment. *Proc. London Math. Soc.* 115, 5 (2017), 1118–1148. https://doi.org/10.1112/plms.12064

[19] Martin Hyland, Gordon Plotkin, and John Power. 2006. Combining Effects: Sum and Tensor. *Theor. Comput. Sci.* 364, 2 (2006), 254–269. https://doi.org/10.1016/j.tcs.2006.03.013

[20] Martin Hyland and Andrea Schalk. 2003. Glueing and Orthogonality for Models of Linear Logic. *Theor. Comput. Sci.* 294, 1–2 (2003), 181–231. https://doi.org/10.1016/s0304-3975(01)00241-9

[21] Mauro Jaskelioff and Eugenio Moggi. 2010. Monad Transformers as Monoid Transformers. *Theor. Comput. Sci.* 411, 51–52 (2010), 4441–4466. https://doi.org/10.1017/s0956796810000122

[22] Shin-ya Katsumata, Exequiel Rivas, and Tarmo Uustalu. 2019. Interaction Laws of Monads and Comonads. arXiv eprint arXiv:1912.13477. https://arxiv.org/abs/1912.13477

[23] Fosco Loregian. 2015. This Is the (Co)end, My Only (Co)friend. arXiv preprint 1501.02503. https://arxiv.org/abs/1501.02503

[24] Saunders Mac Lane. 1978. *Categories for the Working Mathematician, 2nd ed.* Graduate Texts in Mathematics, Vol. 5. Springer, New York. https://doi.org/10.1007/978-1-4757-4721-8

[25] Rasmus Ejlers Møgelberg and Sam Staton. 2014. Linear Usage of State. *Log. Methods Comput. Sci.* 10, 1, Article 17 (2014), 52 pages. https://doi.org/10.2168/lmcs-10(1:17)2014

[26] Eugenio Moggi. 1989. Computational Lambda-calculus and Monads. In *Proc. of 4th Ann. Symp. on Logic in Computer Science, LICS '89*. IEEE Press, Los Alamitos, CA, 14–23. https://doi.org/10.1109/lics.1989.39155

[27] Frank Joseph Oles. 1982. *A Category-Theoretic Approach to the Semantics of Programming Languages.* Ph.D. Dissertation. Syracuse University. http://www.cs.cmu.edu/afs/cs.cmu.edu/project/fox-19/member/jcr/www/FrankOlesThesis.pdf

[28] Dirk Pattinson and Lutz Schröder. 2015. Sound and Complete Equational Reasoning over Comodels. *Electron. Notes Theor. Comput. Sci.* 319 (2015), 315–331. https://doi.org/10.1016/j.entcs.2015.12.019

[29] Dusko Pavlovic. 1997. Chu I: Cofree Equivalences, Dualities and $\star$-Autonomous Categories. *Math. Struct. Comput. Sci.* 7, 1 (1997), 49–73. https://doi.org/10.1017/S0960129596002046

[30] Gordon Plotkin and John Power. 2002. Notions of Computation Determine Monads. In *Proc. of 5th Int. Conf. on Foundations of Software Science and Computation Structures, FoSSaCS 2002*, Mogens Nielsen and Uffe Engberg (Eds.). Lect. Notes in Comput. Sci., Vol. 2303. Springer, Berlin, Heidelberg, 342–356. https://doi.org/10.1007/3-540-45931-6_24

[31] Gordon Plotkin and John Power. 2003. Algebraic Operations and Generic Effects. *Appl. Categ. Struct.* 11, 1 (2003), 69–94. https://doi.org/10.1023/a:1023064908962

[32] Gordon Plotkin and John Power. 2008. Tensors of Comodels and Models for Operational Semantics. *Electron. Notes Theor. Comput. Sci.* 218 (2008), 295–311. https://doi.org/10.1016/j.entcs.2008.10.018

[33] Gordon D. Plotkin and Matija Pretnar. 2013. Handling Algebraic Effects. *Log. Methods Comput. Sci.* 9, 4, Article 23 (2013), 36 pages. https://doi.org/10.2168/lmcs-9(4:23)2013

[34] Hans-E. Porst. 2018. Hopf Monoids in Varieties. *Algebra Universalis* 79, Article 18 (2018), 28 pages. https://doi.org/10.1007/s00012-018-0500-5

[35] Hans-E. Porst and Ross Street. 2016. Generalizations of the Sweedler Dual. *Appl. Categ. Struct.* 24, 5 (2016), 619–647. https://doi.org/10.1007/s10485-016-9450-2

[36] John Power and Olha Shkaravska. 2004. From Comodels to Coalgebras: State and Arrays. *Electron. Notes Theor. Comput. Sci.* 106 (2004), 297–314. https://doi.org/10.1016/j.entcs.2004.02.041

[37] Exequiel Rivas and Mauro Jaskelioff. 2017. Notions of Computation as Monoids. *J. Funct. Program.* 27, Article e21 (2017), 42 pages. https://doi.org/10.1017/s0956796817000132

[38] Moss E. Sweedler. 1969. *Hopf Algebras.* W. A. Benjamin, New York.

[39] Bernardo Toninho, Luís Caires, and Frank Pfenning. 2010. Session Types as Intuitionistic Linear Propositions. In *Proc. of 21th Int. Conf.*

on Concurrency Theory, CONCUR 2010, Paul Gastin and François Laroussinie (Eds.). Lect. Notes in Comput. Sci., Vol. 6269. Springer, Berlin, Heidelberg, 222–236. https://doi.org/10.1007/978-3-642-15375-4_16

[40] Tarmo Uustalu. 2003. Generalizing Substitution. *Theor. Inf. Appl.* 37, 4 (2003), 315–336. https://doi.org/10.1051/ita:2003022

[41] Tarmo Uustalu. 2015. Stateful Runners of Effectful Computations. *Electron. Notes Theor. Comput. Sci.* 319 (2015), 403–421. https://doi.org/10.1016/j.entcs.2015.12.024

[42] Philip Wadler. 2012. Propositions as Sessions. In *Proc. of 17th ACM SIGPLAN Int. Conf. on Functional Programming, ICFP '12*. ACM Press, New York, 273–286. https://doi.org/10.1145/2364527.2364568