Monads Need Not Be Endofunctors

Thorsten Altenkirch[1], James Chapman[2], and Tarmo Uustalu[2]

[1] School of Computer Science, University of Nottingham
[2] Institute of Cybernetics, Tallinn University of Technology
txa@cs.nott.ac.uk, {james,tarmo}@cs.ioc.ee

Abstract. We introduce a generalisation of monads, called relative monads, allowing for underlying functors between different categories. Examples include finite-dimensional vector spaces, untyped and typed λ-calculus syntax and indexed containers. We show that the Kleisli and Eilenberg-Moore constructions carry over to relative monads and are related to relative adjunctions. Under reasonable assumptions, relative monads are monoids in the functor category concerned and extend to monads, giving rise to a coreflection between monads and relative monads. Arrows are also an instance of relative monads.

1 Introduction

Monads are the most successful programming pattern arising in functional programming. Apart from their use to model a generic notion of effect they also serve as a convenient interface to generalized notions of substitution. Research in the area on the border between category theory and functional programming focusses on unveiling new programming and reasoning constructions similar to monads, such as comonads [18], arrows [9] and idioms (closed functors) [13]. Indeed, especially when working in an expressive and total language with dependent types, such as Agda [3], we can exploit monads not only as a way to structure our programs but also their verification.

The present paper is concerned with a generalisation of monads which arises naturally in dependently typed programming, namely monad-like entities that are not endofunctors. Consider the following example, which arose when implementing notions related to quantum programming, namely finite-dimensional vector spaces [19,4]. (See also Piponi [15] for this and other interesting uses of vector spaces in functional programming.)

Example 1. In quantum computing, we consider complex vector spaces, but for the present development any semiring $(R, +, 0, \times, 1)$ is sufficient. Finite-dimensional vector spaces can be given by:

$$\begin{aligned}
&\mathsf{Vec} \in |\mathbf{Fin}| \to |\mathbf{Set}| \\
&\mathsf{Vec}\, m =_{\mathrm{df}} J_{\mathrm{f}}\, m \to R \\
&\eta \in \mathit{\Pi}_{m \in |\mathbf{Fin}|} J_{\mathrm{f}}\, m \to \mathsf{Vec}\, m \\
&\eta_m\, (i \in \underline{m}) =_{\mathrm{df}} \lambda j \in \underline{m}.\ \text{if } i = j \text{ then } 1 \text{ else } 0 \\
&(-)^* \in \mathit{\Pi}_{m,n \in |\mathbf{Fin}|} (J_{\mathrm{f}}\, m \to \mathsf{Vec}\, n) \to (\mathsf{Vec}\, m \to \mathsf{Vec}\, n) \\
&A^*\, x =_{\mathrm{df}} \lambda j \in \underline{n}. \textstyle\sum_{i \in \underline{m}} x\, i \times A\, i\, j
\end{aligned}$$

L. Ong (Ed.): FOSSACS 2010, LNCS 6014, pp. 297–311, 2010.

Here **Fin** is the category of finite cardinals (the skeletal version of finite sets). The objects are natural numbers $m \in \mathbb{N}$ and the maps between m and n are functions between $\underline{m}$ and $\underline{n}$ where $\underline{m} =_{\mathrm{df}} \{0, 1, \ldots, m-1\}$. By $J_{\mathrm{f}} \in \mathbf{Fin} \to \mathbf{Set}$ we mean the natural embedding $J_{\mathrm{f}}\, m =_{\mathrm{df}} \underline{m}$. The finite summation $\sum$ is just the finite iteration of $+$ over 0. Indeed η_m is just the unit $m \times m$-matrix (alternatively, a function assigning to every coordinate $i \in \underline{m}$ the corresponding unit vector) and $A^*\, x$ corresponds to the product of the matrix A with the vector x, where both matrices and vectors are described as functions.

By the types of its data, the structure $(\mathsf{Vec}, \eta, (-)^*)$ looks suspiciously like a monad, except that **Fin** is not **Set** and in the types for η and $(-)^*$ we have used the embedding J_{f} to repair the mismatch. It is easy to verify that the structure also satisfies the standard monad laws, modulo the same discrepancy.

The category of finite-dimensional vector spaces arises as a kind of Kleisli category. Its objects are $m \in \mathbb{N}$, understood as finite coordinate systems (describing vector spaces), and its morphisms are functions $J_{\mathrm{f}} m \to \mathsf{Vec}\, n$, i.e., matrices (describing linear transformations).

The structure cannot generally be pushed to a monad on **Set**. $(-)^*$ requires that we can sum over a set. Summation over general index sets is not available, if R is just a semiring. Also, in a constructive setting, η requires that the set has a decidable equality, which is not the case for general sets.

We view Vec as a *relative monad* on the embedding $J_{\mathrm{f}} \in \mathbf{Fin} \to \mathbf{Set}$. Other examples of relative monads include untyped and simply typed λ-terms and the notions of indexed functors and indexed containers as developed in [14].

Overview of the paper. In Sect. 2 we develop the notion of relative monads on a functor $J \in \mathbb{J} \to \mathbb{C}$, showing that they arise from relative adjunctions, and generalize Kleisli and Eilenberg-Moore constructions to relative monads.

Since monads on $\mathbb{C}$ correspond to monoids in the endofunctor category $[\mathbb{C}, \mathbb{C}]$, a natural question is whether a relative monad on $\mathbb{J}$ gives rise to a monoid in the category $[\mathbb{J}, \mathbb{C}]$. If $\mathbb{J}$ is small and $\mathbb{C}$ is cocomplete (e.g., **Set**), the left Kan extension along J exists and give rise to a lax monoidal structure where the unit is J and the tensor is given by $F \cdot^J G =_{\mathrm{df}} \mathrm{Lan}_J\, F \cdot G$. Indeed, relative monads give rise to lax monoids in this lax setting (Sect. 3).

Going further, we identify conditions on the functor J, under which the lax monoidal structure induced by Lan_J is properly monoidal. In this case, we obtain a proper monoid in the category of functors. Moreover, relative monads extend to monads via Lan_J and we get a coreflection between monads and relative monads (Sect. 4). In the example of vector spaces, $\mathrm{Lan}_J\, \mathsf{Vec}$ is the monad whose Kleisli category is that of vector spaces over general sets of coordinates where a vector over an infinite set of coordinates may only have finitely many non-zero components. However, it is worthwhile not to ignore the non-endofunctor case, because frequently this is the structure we actually want to use. E.g., in quantum computing we are interested in dagger compact closed categories [2].

Finally, we show that arrows are relative monads (Sect. 5) on the Yoneda embedding. This leads to the, maybe surprising, outcome that while arrows generalize ordinary monads, they are actually a special case of relative monads.

Related work The untyped λ-calculus syntax as has been identified as a monoid in $[\mathbf{Fin}, \mathbf{Set}]$ by Fiore et al. [7]. Heunen and Jacobs [8] have shown that arrows on $\mathbb{C}$ are actually monoids in the category $[\mathbb{C}^{\mathrm{op}} \times \mathbb{C}, \mathbf{Set}]$ of endoprofunctors; Jacobs et al. have proved the Freyd construction of [16] is, in a good sense, the Kleisli construction for arrows. Spivey [17] has studied a generalization of monads, which differs from ours, but is similar in spirit and related (see Conclusion). That the monoid nature of monads is important in developing applications of monads, was recently shown by Jaskelioff in his work on modular monad transformers [11].

Notation. We will be using a mixture of categorical and type-theoretic notation. In particular we will be using λ-calculus notation for defining functions (maps in **Set** or subcategories). Customarily for both category theory and type theory, we often hide some arguments of patterns and function applications (normally subscripted arguments, e.g., an object a natural transformation is applied to).

We write $|\mathbb{C}|$ for the objects of $\mathbb{C}$ and $\mathbb{C}(X, Y)$ for the homsets. Given categories $\mathbb{C}$,$\mathbb{D}$ we write the functor category as $[\mathbb{C}, \mathbb{D}]$. We write id, $\circ$ for the identities and composition of maps and I, $\cdot$ for the identities and composition of functors.

2 Relative Monads and Relative Adjunctions

We start by defining relative monads. Then we give some examples and show how the theory of ordinary monads carries over to the relative case.

2.1 Relative Monads

Rather than being defined for a category $\mathbb{C}$ like a monad, a relative monad is defined for a functor J between two categories $\mathbb{J}$ and $\mathbb{C}$.

Definition 1. *A (Manes-style [12])* relative monad *on a functor $J : \mathbb{J} \to \mathbb{C}$ is given by*

- *an object mapping $T \in |\mathbb{J}| \to |\mathbb{C}|$,*
- *for any $X \in |\mathbb{J}|$, a map $\eta_X \in \mathbb{C}(J X, T X)$ (the* unit*),*
- *for any $X, Y \in |\mathbb{J}|$ and $k \in \mathbb{C}(J X, T Y)$, a map $k^* \in \mathbb{C}(T X, T Y)$ (the* Kleisli extension*)*

satisfying the conditions

- *for any $X, Y \in |\mathbb{J}|$, $k \in \mathbb{C}(J X, T Y)$, $k = k^* \circ \eta$,*
- *for any $X \in |\mathbb{J}|$, $\eta^*_X = \mathrm{id}_{TX} \in \mathbb{C}(T X, T X)$,*
- *for any $X, Y, Z \in |\mathbb{J}|$, $k \in \mathbb{C}(J X, T Y)$, $\ell \in \mathbb{C}(J Y, T Z)$, $(\ell^* \circ k)^* = \ell^* \circ k^*$.*

The data and laws of a relative monad are exactly as those of a monad, except that $\mathbb{C}$ has become $\mathbb{J}$ in some places and, to ensure type-compatibility, some occurrences of J have been inserted.

Although this is not stated in the axioms, they imply that T is functorial: $T \in \mathbb{J} \to \mathbb{C}$. Indeed, for $X, Y \in |\mathbb{J}|$, $f \in \mathbb{J}(X, Y)$, we can define a map

$T\,f \in \mathbb{C}\,(TX, TY)$ by $T\ f =_{\mathrm{df}} (\eta \circ Jf)^*$ and this satisfies the functor laws. Also, η and $(-)^*$ are natural.

A definition of relative monads based on a multiplication μ rather than a Kleisli extension $(-)^*$ is not immediately available: the simple functor composition $T \cdot T$ is not well-typed. In the next section, we will show that a suitable notion of functor composition is available under a condition.

Clearly, monads are a special case of relative monads via $\mathbb{J} =_{\mathrm{df}} \mathbb{C}$, $J =_{\mathrm{df}} I_{\mathbb{C}}$.

For general $\mathbb{J}$, $\mathbb{C}$ and J, we always have that $T\,X =_{\mathrm{df}} J\,X$ is a relative monad with $\eta_X =_{\mathrm{df}} \mathrm{id}_{JX}$ and $k^* =_{\mathrm{df}} k$. A whole class of examples of relative monads on J is given by restricting monads on $\mathbb{C}$ (the relative monad J arises from restricting the monad $I_{\mathbb{C}}$).

Theorem 1. *For any $J \in \mathbb{J} \to \mathbb{C}$, a monad $(T, \eta, (-)^*)$ on $\mathbb{C}$ restricts to a relative monad $(T^\flat, \eta^\flat, (-)^{(*^\flat)})$ on J, defined by $T^\flat\,X =_{\mathrm{df}} T\,(J\,X)$, $\eta^\flat_X =_{\mathrm{df}} \eta_{J\,X}$, $k^{(*^\flat)} =_{\mathrm{df}} k^*$.*

As a first truly non-trivial example, we saw the relative monad of finite-dimensional vector spaces in the introduction. Here are some further examples.

Example 2. The syntax of untyped (but well-scoped) λ-calculus is a relative monad on $J_{\mathrm{f}} \in \mathbf{Fin} \to \mathbf{Set}$, as the finite-dimensional vector spaces relative monad, i.e., we have $\mathbb{J} =_{\mathrm{df}} \mathbf{Fin}$, $\mathbb{C} =_{\mathrm{df}} \mathbf{Set}$, $J =_{\mathrm{df}} J_{\mathrm{f}}$. We view $\mathbf{Fin}$ as the category of nameless untyped contexts. The set of untyped λ-terms $\mathsf{Lam}\,\Gamma$ over a context Γ satisfies the isomorphism

$$\mathsf{Lam}\,\Gamma \cong J_{\mathrm{f}}\,\Gamma + \mathsf{Lam}\,\Gamma \times \mathsf{Lam}\,\Gamma + \mathsf{Lam}\,(1 + \Gamma)$$

The summands correspond to variables from the context (seen as terms), applications, and abstractions (their bodies are terms over an extended context). The functor $\mathsf{Lam} \in \mathbf{Fin} \to \mathbf{Set}$ is defined as the carrier of the initial algebra of the functor $F \in [\mathbf{Fin}, \mathbf{Set}] \to [\mathbf{Fin}, \mathbf{Set}]$ defined by

$$F\,G\,\Gamma =_{\mathrm{df}} J_{\mathrm{f}}\,\Gamma + G\,\Gamma \times G\,\Gamma + G\,(1 + \Gamma)$$

Lam is a relative monad. The unit $\eta \in J_{\mathrm{f}}\,\Gamma \to \mathsf{Lam}\,\Gamma$ is given by variables-as-terms and the Kleisli extension takes a finite substitution rule $k \in J_{\mathrm{f}}\,\Gamma \to \mathsf{Lam}\,\Delta$ to the corresponding substitution function $k^* \in \mathsf{Lam}\,\Gamma \to \mathsf{Lam}\,\Delta$.

This example was described as a relative monad (under the name Kleisli structure) by Altenkirch and Reus [5]. Fiore et al. [7] described it as a monoid in a monoidal structure on $[\mathbf{Fin}, \mathbf{Set}]$. Their account of this example is an instance of our general description of relative monads as monoids from Section 4.

Example 3. Typed λ-terms form a relative monad in a similar fashion. Let Ty be the set of types (over some base types), which we see as a discrete category. We take $\mathbb{J}$ to be $\mathbf{Fin} \downarrow \mathsf{Ty}$, which is the category whose objects are pairs (Γ, ρ) where $\Gamma \in |\mathbf{Fin}|$ and $\rho \in \Gamma \to \mathsf{Ty}$ (typed contexts) and maps from (Γ, ρ) to (Γ', ρ') are maps $f \in \mathbf{Fin}\,(\Gamma, \Gamma')$ such that $\rho = \rho' \circ f$ (typed context maps).

We further take $\mathbb{C}$ to be the functor category $[\mathsf{Ty}, \mathbf{Set}]$ and let $J \in \mathbf{Fin} \downarrow \mathsf{Ty} \to [\mathsf{Ty}, \mathbf{Set}]$ be the natural embedding defined by $J\,(\Gamma, \rho)\;\sigma =_{\mathrm{df}} \{x \in \Gamma \mid \rho\, x = \sigma\}$.

Now, for $(\Gamma, \rho) \in |\mathbf{Fin} \downarrow \mathsf{Ty}|$ and $\sigma \in \mathsf{Ty}$, the set of typed λ-terms $\mathsf{TyLam}\,(\Gamma, \rho)\,\sigma$ has to satisfy the isomorphism

$$\begin{aligned}\mathsf{TyLam}\,(\Gamma, \rho)\,\sigma \cong\; & J\,(\Gamma, \rho)\,\sigma \\ & + \Sigma_{\tau \in \mathsf{Ty}} \mathsf{TyLam}\,(\Gamma, \rho)\,(\tau \Rightarrow \sigma) \times \mathsf{TyLam}\,(\Gamma, \rho)\,\tau \\ & + \text{if } \sigma \text{ is of the form } \tau \Rightarrow \tau' \text{ then } \mathsf{TyLam}\,(1 + \Gamma, \begin{bmatrix} \mathrm{inl}\, * \mapsto \tau \\ \mathrm{inr}\, x \mapsto \rho\, x \end{bmatrix})\,\tau'\end{aligned}$$

The functor $\mathsf{TyLam} \in \mathbf{Fin} \downarrow \mathsf{Ty} \to [\mathsf{Ty}, \mathbf{Set}]$ is given by an initial algebra. It is a monad on J, with the unit and Kleisli extension given by variables-as-terms and substitution, like in the case of Lam. Fiore et al. [6] studied TyLam as a monoid in $[\mathbf{Fin} \downarrow \mathsf{Ty}, [\mathsf{Ty}, \mathbf{Set}]]$.

Note that choosing $\mathbb{J}$ to be $[\mathsf{Ty}, \mathbf{Fin}]$ rather than $\mathbf{Fin} \downarrow \mathsf{Ty}$ would have given contexts possibly supported by infinitely many types: in every type there are finitely many variables, but the total number of variables can be infinite.

Example 4. Morris and Altenkirch [14] investigated generalization of the notion of containers [1] to a dependently typed setting and used it to show that strictly positive families can be reduced to W-types. Relative monads played a central role in this development.

Let $\mathsf{U} \in \mathbf{Set}$ together with $\mathsf{El} \in \mathsf{U} \to \mathbf{Set}$ be a universe of small sets. This induces a category $\mathbf{U}$ with $|\mathbf{U}| =_{\mathrm{df}} \mathsf{U}$ and $\mathbf{U}\,(a, b) =_{\mathrm{df}} \mathsf{El}\, a \to \mathsf{El}\, b$. The functor $J_{\mathbf{U}} \in \mathbf{U} \to \mathbf{Cat}$ is given by $J_{\mathbf{U}}\, a =_{\mathrm{df}} \mathsf{El}\, a$ on objects and the identity on maps (viewing $\mathsf{El}\, a$ as a discrete category). We assume that $\mathbf{U}$ is locally cartesian closed, i.e., the universe is closed under dependent product and function types as well as equality types.

As ordinary containers represent endofunctors on $\mathbf{U}$ (or any other locally cartesian closed category), indexed containers represent functors from a slice over a given $a \in U$, we define the category $\mathsf{IF}\, a$ of indexed functors over a by $\mathsf{IF}\, a =_{\mathrm{df}} [[\mathsf{El}\, a, \mathbf{U}], \mathbf{U}]$. The functor $\mathsf{IF} \in \mathbf{U} \to \mathbf{Cat}$ is a relative monad on $J_{\mathbf{U}}$. The unit $\eta_a \in J_{\mathbf{U}}\, a \to \mathsf{IF}\, a$ is defined by $\eta_a\, x =_{\mathrm{df}} \lambda f.\, f\, x$ and the Kleisli extension $k^* \in \mathsf{IF}\, a \to \mathsf{IF}\, b$ of $k \in J_{\mathbf{U}}\, a \to \mathsf{IF}\, b$ is defined by $k^*\, G\, f = G\,(\lambda x.k\, x\, f)$. The definitions clearly resemble the continuation monad apart from the size issue.

The main result of [14] was that strictly positive families (SPF) can be interpreted as indexed functors by via indexed containers (IC). Just as IF, both SPF and IC are relative monads on $J_{\mathbf{U}}$ and the interpretations preserve this structure, i.e., are relative monad maps.

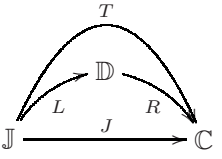
2.2 Relative Adjunctions

As ordinary monads are intimately related to adjunctions, relative monads are related to a corresponding generalization of adjunctions.

Definition 2. *A* relative adjunction *between $J \in \mathbb{J} \to \mathbb{C}$ and $\mathbb{D}$ is given by two functors $L \in \mathbb{J} \to \mathbb{D}$ and $R \in \mathbb{D} \to \mathbb{C}$, and a natural isomorphism $\phi \in \mathbb{C}\,(J\, X, R\, Y) \cong \mathbb{D}\,(L\, X, Y)$.*

As expected, ordinary adjunctions are a special case of relative adjunctions with $\mathbb{J} =_{\mathrm{df}} \mathbb{C}$, $J =_{\mathrm{df}} I$. Just like any adjunction defines a monad, relative adjunctions define relative monads.

Theorem 2. *Any relative adjunction (L, R, ϕ) between a functor $J \in \mathbb{J} \to \mathbb{C}$ and category $\mathbb{D}$ gives rise to a relative monad, defined by $T\,X =_{\mathrm{df}} R\,(L\,X)$, $\eta_X =_{\mathrm{df}} \phi^{-1}\,(\mathrm{id}_{L\,X})$, $k^* =_{\mathrm{df}} R\,(\phi\,k)$.*



If a relative monad T on J is related to a relative adjunction (L, R, ϕ) between J and some category $\mathbb{D}$ in the above way, we call the relative adjunction a *splitting* of the relative monad via $\mathbb{D}$.

2.3 Kleisli and Eilenberg-Moore Constructions

For monads we know that they split into an adjunction in two canonical ways: the Kleisli and Eilenberg-Moore constructions. Moreover, the splittings form a category where the Kleisli and EM splittings are the initial and terminal objects. We shall now establish that the same holds in the relative situation.

The Kleisli category $\mathbf{Kl}(T)$ of a relative monad T has as objects the objects of $\mathbb{J}$ and as maps between X, Y the maps between $J\,X$, $T\,Y$ of $\mathbb{C}$: $|\mathbf{Kl}(T)| =_{\mathrm{df}} |\mathbb{J}|$ and $\mathbf{Kl}(T)\,(X, Y) =_{\mathrm{df}} \mathbb{C}\,(J\,X, T\,Y)$. The identity and composition (we denote them by id^T, $\circ^T$) are defined by $\mathrm{id}^T_X =_{\mathrm{df}} \eta_X$ and $\ell \circ^T k =_{\mathrm{df}} \ell^* \circ k$.

The Kleisli relative adjunction between J and $\mathbf{Kl}(T)$ is defined by $L\,X =_{\mathrm{df}} X$, $L\,f =_{\mathrm{df}} \eta \circ J f$ (note that L is identity-on-objects), $R\,X =_{\mathrm{df}} T\,X$, $R\,k =_{\mathrm{df}} k^*$ and ϕ is identity. This relative adjunction is a splitting: it is immediate that $R\,(L\,X) = T\,X$, $\eta_X = \phi^{-1}\,(\mathrm{id}^T_{L\,X})$, $k^* = R\,(\phi\,k)$.

The Eilenberg-Moore (EM) category $\mathbf{EM}(T)$ is given by EM-algebras and EM-algebra maps of the relative monad T. Since the usual definition of an EM-algebra refers to μ, which is not immediately available, we generalize a version based on $(-)^*$. For ordinary monads this is equivalent to the standard definition.

Definition 3. *An EM-*algebra *of a relative monad T on $J \in \mathbb{J} \to \mathbb{C}$ is given by an object $X \in |\mathbb{C}|$ and, for any $Z \in |\mathbb{J}|$, a map function $\chi \in \mathbb{C}\,(J\,Z, X) \to \mathbb{C}\,(T\,Z, X)$, satisfying the conditions*

- *for any $Z \in |\mathbb{J}|$, $f \in \mathbb{C}\,(J\,Z, X)$, $f = \chi\,f \circ \eta$,*
- *for any $Z, W \in |\mathbb{J}|$, $k \in \mathbb{C}\,(J\,Z, T\,W)$, $f \in \mathbb{C}\,(J\,W, X)$, $\chi\,(\chi\,f \circ k) = \chi\,f \circ k^*$.*

These conditions ensure, among other things, that χ is natural.

*An EM-*algebra map *from (X, χ) to (Y, υ) is a map $h \in \mathbb{C}\,(X, Y)$ satisfying*

- *for any $Z \in |\mathbb{J}|$, $f \in \mathbb{C}\,(J\,Z, X)$, $h \circ \chi\,f = \upsilon\,(h \circ f)$.*

The identity and composition of $\mathbf{EM}(T)$ are inherited from $\mathbb{C}$.

The Eilenberg-Moore relative adjunction between J and $\mathbf{EM}(T)$ is defined by $L\,Y =_{\mathrm{df}} (T\,Y, (-)^*)$, $L\,f =_{\mathrm{df}} T\,f$, $R\,(X, \chi) =_{\mathrm{df}} X$, $R\,h =_{\mathrm{df}} h$ (so R is identity-on-maps), $\phi_{X,(Y,\upsilon)}\,f =_{\mathrm{df}} \upsilon\,f$ and $\phi^{-1}_{X,(Y,\upsilon)}\,h =_{\mathrm{df}} h \circ \eta_X$. This is also a splitting.

Theorem 3. *The splittings of a relative monad T on $J \in \mathbb{J} \to \mathbb{C}$ form a category. An object is given by a category $\mathbb{D}$ and an adjunction (L, R, ϕ) splitting T via $\mathbb{D}$. A splitting morphism between $(\mathbb{D}, L, R, \phi)$ and (D', L', R', ϕ') is a functor $V \in \mathbb{D} \to \mathbb{D}'$ such that $V \cdot L = L'$, $R = R' \cdot V$, and $V\,\phi_{X,Y} = \phi'_{X,V\,Y}$. The Kleisli construction is the initial and the Eilenberg-Moore construction the terminal splitting.*

Example 5. The Kleisli category of Vec has as objects the objects of **Fin** understood as finite coordinate systems (describing vector spaces). The maps are maps $J_{\mathrm{f}}\,m \to \mathsf{Vec}\,n$, i.e., $m \times n$-matrixes (describing linear transformations). The identities are the unit $m \times m$-matrices, the composition is multiplication of matrices.

Example 6. The Kleisli category of Lam has as objects the objects of **Fin** understood as untyped contexts. The maps are maps $J_{\mathrm{f}}\,\Gamma \to \mathsf{Lam}\,\Delta$, i.e., substitution rules (assignments of terms over Δ to the variables in Γ). The identities are the trivial substitution rules. The composition is composition of substitution rules.

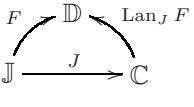
3 Relative Monads as Lax Monoids

A monad on $\mathbb{C}$ is the same as a monoid in the endofunctor category $[\mathbb{C}, \mathbb{C}]$. It has a monoidal structure given by the identity functor I and composition of functors $\cdot$, which are strictly unital and associative. A monad can be specified by an object $T \in |[\mathbb{C}, \mathbb{C}]|$ and maps $\eta \in [\mathbb{C}, \mathbb{C}]\,(I, T)$ and $\mu \in [\mathbb{C}, \mathbb{C}]\,(T \cdot T, T)$ satisfying the laws of a monoid in the strict monoidal category $([\mathbb{C}, \mathbb{C}], I, \cdot)$.

Can we similarly define a relative monad on $J \in \mathbb{J} \to \mathbb{C}$ as a monoid in the functor category $[\mathbb{J}, \mathbb{C}]$? This requires a monoidal structure on $[\mathbb{J}, \mathbb{C}]$, ideally similar to that on $[\mathbb{C}, \mathbb{C}]$. The functor J is a good candidate for the unit, but the tensor is problematic, as functors $\mathbb{J} \to \mathbb{C}$ cannot be composed by simple functor composition. We shall use a left Kan extension to overcome the difficulty and obtain a lax monoidal structure where relative monads are lax monoids.

3.1 Left Kan Extensions

Left Kan extensions are one of the two canonical constructions for extending functors. The left Kan extension along $J \in \mathbb{J} \to \mathbb{C}$ extends functors $\mathbb{J} \to \mathbb{D}$ to functors $\mathbb{C} \to \mathbb{D}$.



It is defined as the left adjoint (if it exists) of the restriction functor $- \cdot J \in [\mathbb{C}, \mathbb{D}] \to [\mathbb{J}, \mathbb{D}]$. By definition, it is given by a functor $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{D}] \to [\mathbb{C}, \mathbb{D}]$ and a natural isomorphism

$$[\mathbb{J}, \mathbb{D}]\,(F, G \cdot J) \cong [\mathbb{C}, \mathbb{D}]\,(\mathrm{Lan}_J\, F, G)$$

While it is possible to work directly with this definition of left Kan extension, we use an alternative definition, based on the coend formula

$$\mathrm{Lan}_J\, F\, X \cong \int^{Y \in |\mathbb{J}|} \mathbb{C}\,(J\, Y, X) \bullet F\, Y$$

Accordingly, we take that a left Kan extension of a functor $F \in \mathbb{J} \to \mathbb{D}$ along $J \in \mathbb{J} \to \mathbb{C}$ to be given by

- an object function $\mathrm{Lan}_J\, F \in |\mathbb{C}| \to |\mathbb{D}|$,
- for any $X \in |\mathbb{C}|$, a natural transformation $\iota_{F,X} \in [\mathbb{J}^{\mathrm{op}}, \mathbf{Set}]\,(\mathbb{C}\,(J -, X), \mathbb{D}\,(F -, \mathrm{Lan}_J\, F\, X))$,
- for any $X \in |\mathbb{C}|$, $Y \in |\mathbb{D}|$ and $\theta \in [\mathbb{J}^{\mathrm{op}}, \mathbf{Set}]\,(\mathbb{C}\,(J -, X), \mathbb{D}\,(F -, Y))$, a map $[\theta] \in \mathbb{D}\,(\mathrm{Lan}_J\, F\, X, Y)$.

satisfying the conditions $[\theta] \circ \iota\, g = \theta\, g$, $[\iota] = \mathrm{id}$ and $f \circ [\theta] = [\lambda g.\, f \circ \theta\, g]$.

Left Kan extensions $\mathrm{Lan}_J\, F\, X$ are functorial in both arguments F and X, i.e., $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{D}] \to [\mathbb{C}, \mathbb{D}]$. For any $F \in |[\mathbb{J}, \mathbb{D}]|$, $X, Y \in |\mathbb{C}|$, $f \in \mathbb{C}\,(X, Y)$,

$$\begin{aligned} &\mathrm{Lan}_J\, F\, f \in \mathbb{D}\,(\mathrm{Lan}_J\, F\, X, \mathrm{Lan}_J\, F\, Y) \\ &\mathrm{Lan}_J\, F\, f =_{\mathrm{df}} [\lambda g.\, \iota\,(f \circ g)] \end{aligned}$$

And for any $F, G \in |[\mathbb{J}, \mathbb{D}]|$, $\tau \in [\mathbb{J}, \mathbb{D}]\,(F, G)$, $X \in |\mathbb{C}|$, we have

$$\begin{aligned} &\mathrm{Lan}_J\, \tau\, X \in \mathbb{D}\,(\mathrm{Lan}_J\, F\, X, \mathrm{Lan}_J\, G\, X) \\ &\mathrm{Lan}_J\, \tau\, X =_{\mathrm{df}} [\lambda g.\, \iota\, g \circ \tau] \end{aligned}$$

In general $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{D}] \to [\mathbb{C}, \mathbb{D}]$ exists, if $\mathbb{J}$ is small and $\mathbb{D}$ is cocomplete.

3.2 $[\mathbb{J}, \mathbb{C}]$ Is Lax Monoidal

If $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{C}] \to [\mathbb{C}, \mathbb{C}]$ exists, we can turn any functor $F \in |[\mathbb{J}, \mathbb{C}]|$ to one in $|[\mathbb{C}, \mathbb{C}]|$. Hence we can define a composition-like operation

$$\begin{aligned} &(\cdot^J) \in |[\mathbb{J}, \mathbb{C}]| \times |[\mathbb{J}, \mathbb{C}]| \to |[\mathbb{J}, \mathbb{C}]| \\ &F \cdot^J G =_{\mathrm{df}} \mathrm{Lan}_J\, F \;\cdot\; G \end{aligned}$$

This is our candidate for the tensor on $[\mathbb{J}, \mathbb{C}]$. We also need the unital and associative laws. We define several families of maps indexed by $X \in |\mathbb{C}|$:

$$\begin{aligned} &\overline{\lambda}_X \in \mathbb{C}\,(\mathrm{Lan}_J\, J\, X, X) \\ &\overline{\lambda}_X =_{\mathrm{df}} [\lambda g.\, g] \\ &\overline{\overline{\alpha}}_{F,G,X} \in \mathbb{C}\,(\mathrm{Lan}_J\,(F \cdot G)\, X, F\,(\mathrm{Lan}_J\, G\, X)) \\ &\overline{\overline{\alpha}}_{F,G,X} =_{\mathrm{df}} [\lambda g.\, F\,(\iota\, g)] \\ &\overline{\alpha}_{F,G,X} \in \mathbb{C}\,(\mathrm{Lan}_J\,(\mathrm{Lan}_J\, F \cdot G)\, X, \mathrm{Lan}_J\, F\,(\mathrm{Lan}_J\, G\, X)) \\ &\overline{\alpha}_{F,G,X} =_{\mathrm{df}} \overline{\overline{\alpha}}_{\mathrm{Lan}_J F, G} = [\lambda g.\, [\lambda g'.\, \iota\,(\iota\, g \circ g')]] \end{aligned}$$

All these families are natural in X, hence maps in $|[\mathbb{C}, \mathbb{C}]|$.

From these we further define our candidate unital and associative laws.

$$\begin{aligned}
&\rho_F \in [\mathbb{J}, \mathbb{C}]\,(F, F \cdot^J J)\\
&\rho_F =_{\mathrm{df}} \iota\,\mathrm{id}\\
&\lambda_F \in [\mathbb{J}, \mathbb{C}]\,(J \cdot^J F, F)\\
&\lambda_F =_{\mathrm{df}} \overline{\lambda} \cdot F\\
&\alpha_{F,G,H} \in [\mathbb{J}, \mathbb{C}]\,((F \cdot^J G) \cdot^J H, F \cdot^J (G \cdot^J H))\\
&\alpha_{F,G,H} =_{\mathrm{df}} \overline{\alpha}_{F,G} \cdot H
\end{aligned}$$

It turns out that the data so defined provide a structure that is almost monoidal, but not quite. It is lax monoidal: λ, ρ, α are generally not isomorphisms. In the next section we will identify conditions on J that enable us to construct the inverses, turning the lax monoidal structure into properly monoidal.

Theorem 4. *If* $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{C}] \to [\mathbb{C}, \mathbb{C}]$ *exists, then* $([\mathbb{J}, \mathbb{C}], J, \cdot^J, \lambda, \rho, \alpha)$ *is a lax monoidal category, i.e.,* $\cdot^J$ *is functorial,* λ, ρ, α *are natural and the following diagrams commute:*

$$\begin{array}{ccc}
J \cdot^J J & \xrightarrow{\rho_J} & J \\
& \searrow^{\lambda_J} & \| \\
& & J
\end{array}
\qquad
\begin{array}{ccc}
(F \cdot^J J) \cdot^J G & \xrightarrow{\alpha_{F,J,G}} & F \cdot^J (J \cdot^J G) \\
{\scriptstyle \rho_F \cdot^J G}\uparrow & & \downarrow{\scriptstyle F \cdot^J \lambda_G} \\
F \cdot^J G & = & F \cdot^J G
\end{array}$$

$$\begin{array}{ccc}
(J \cdot^J F) \cdot^J G & \xrightarrow{\alpha_{J,F,G}} & J \cdot^J (F \cdot^J G) \\
& {\scriptstyle \lambda_F \cdot^J G}\searrow \quad \swarrow{\scriptstyle \lambda_{F \cdot^J G}} & \\
& F \cdot^J G &
\end{array}
\qquad
\begin{array}{ccc}
(F \cdot^J G) \cdot^J J & \xrightarrow{\alpha_{F,G,J}} & F \cdot^J (G \cdot^J J) \\
& {\scriptstyle \rho_{F \cdot^J G}}\nwarrow \quad \nearrow{\scriptstyle F \cdot^J \rho_G} & \\
& F \cdot^J G &
\end{array}$$

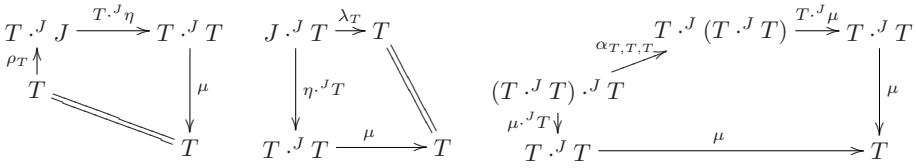
$$\begin{array}{ccccc}
(F \cdot^J (G \cdot^J H)) \cdot^J K & & \xrightarrow{\alpha_{F,G \cdot^J H,K}} & & F \cdot^J ((G \cdot^J H) \cdot^J K) \\
{\scriptstyle \alpha_{F,G,H} \cdot^J K}\uparrow & & & & \downarrow{\scriptstyle F \cdot^J \alpha_{G,H,K}} \\
((F \cdot^J G) \cdot^J H) \cdot^J K & \xrightarrow{\alpha_{F \cdot^J G,H,K}} & (F \cdot^J G) \cdot^J (H \cdot^J K) & \xrightarrow{\alpha_{F,G,H \cdot^J K}} & F \cdot^J (G \cdot^J (H \cdot^J K))
\end{array}$$

3.3 Relative Monads Are the Same as Lax Monoids in $[\mathbb{J}, \mathbb{C}]$

With a lax monoidal structure present on the functor category $[\mathbb{J}, \mathbb{C}]$, we should expect that relative monads on J are the same thing as lax monoids in this structure, generalizing the case of ordinary monads on $\mathbb{C}$ and the strict monoidal structure on the endofunctor category $[\mathbb{C}, \mathbb{C}]$. This is indeed the case.

Theorem 5. *Assume that* $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{C}] \to [\mathbb{C}, \mathbb{C}]$ *exists.*

1. *Given a relative monad* $(T, \eta, (-)^*)$ *on* J*, define, for any* $X \in |\mathbb{J}|$*, a map* $\mu_X \in \mathbb{C}\,(\mathrm{Lan}_J T\,(T\,X), T\,X)$ *by* $\mu_X =_{\mathrm{df}} [(-)^*]$*. This is well-defined, since* $(-)^*$ *is natural:* $(-)^* \in [\mathbb{J}^{\mathrm{op}}, \mathbf{Set}]\,(\mathbb{C}\,(J -, T\,X), \mathbb{C}\,(T -, T\,X))$.
   Then (T, η, μ) *is a lax monoid in the lax monoidal category* $([\mathbb{J}, \mathbb{C}], J, \cdot^J, \lambda, \rho, \alpha)$*: we have that* $T \in |[\mathbb{J}, \mathbb{C}]|$*,* $\eta \in [\mathbb{J}, \mathbb{C}]\,(J, T)$ *and* $\mu \in [\mathbb{J}, \mathbb{C}]\,(T \cdot^J T, T)$*, and the following diagrams commute in* $[\mathbb{J}, \mathbb{C}]$*:*



2. *Given a lax monoid* (T, η, μ) *in* $([\mathbb{J}, \mathbb{C}], J, \cdot^J, \lambda, \rho, \alpha)$*, define, for any* $X, Y \in |\mathbb{J}|$*, a function* $(-)^* \in \mathbb{C}\,(J\,X, T\,Y) \to \mathbb{C}\,(T\,X, T\,Y)$ *by* $k^* =_{\mathrm{df}} \mu_Y \circ \iota\, k$*. Then* $(T, \eta, (-)^*)$ *is a relative monad on* J*.*
3. *The above correspondence is bijective.*

The bijective correspondence between relative monads on J and lax monoids in $[\mathbb{J}, \mathbb{C}]$ extends to an equivalence of categories, but we must omit the details here (we have defined neither relative monad maps nor lax monoid maps).

Moreover, just as the availability of $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{C}] \to [\mathbb{C}, \mathbb{C}]$ allows us to define relative monads based on μ rather than $(-)^*$, it also facilitates a more traditional-style definition of EM-algebras; we must omit the details.

4 Well-Behaved Relative Monads

It is somewhat unsatisfactory to obtain that $[\mathbb{J}, \mathbb{C}]$ is just lax monoidal, rather than properly monoidal. This begs the question: would some conditions on J ensure a properly monoidal structure? The answer is affirmative. Mild conditions turn the lax monoidal structure of $[\mathbb{J}, \mathbb{C}]$ into properly monoidal. What is more, the same conditions also make relative monads on J extend to monads on $\mathbb{C}$.

4.1 Well-Behavedness Conditions

We define three well-behavedness conditions on J. They are additional to the existence of $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{C}] \to [\mathbb{C}, \mathbb{C}]$ and require the constituent maps of three canonical families, which are actually natural, to be isomorphisms. For our purposes, these conditions are mild.

Definition 4. $J \in \mathbb{J} \to \mathbb{C}$ *is* well-behaved*, if not only does* $\mathrm{Lan}_J \in [\mathbb{J}, \mathbb{C}] \to [\mathbb{C}, \mathbb{C}]$ *exist, but also the following three conditions hold:*

1. J *is* fully faithful*, i.e., for any* $X, Y \in |\mathbb{J}|$*, there is an inverse to the map*

$$J_{X,Y} \in \mathbb{J}\,(X, Y) \to \mathbb{C}\,(J\,X, J\,Y)$$
$$J_{X,Y}\,f =_{\mathrm{df}} J\,f$$

2. J *is* dense*, i.e., for any* $X, Y \in |\mathbb{C}|$*, there is an inverse to the map*

$$K_{X,Y} \in \mathbb{C}\,(X, Y) \to [\mathbb{J}^{\mathrm{op}}, \mathbf{Set}]\,(\mathbb{C}\,(J\,-, X), \mathbb{C}\,(J\,-, Y))$$
$$K_{X,Y}\,f =_{\mathrm{df}} \lambda g.\, f \circ g$$

i.e. $K \in \mathbb{C} \to [\mathbb{J}^{\mathrm{op}}, \mathbf{Set}]$*, with* $K\,X =_{\mathrm{df}} \mathbb{C}\,(J\,-, X)$*, is fully faithful.*

3. *For any* $F \in \mathbb{J} \to \mathbb{C}$, $X \in |\mathbb{J}|$, $Y \in |\mathbb{C}|$, *there is an inverse to the map*

$$L^F_{X,Y} \in \mathrm{Lan}_J\,(\mathbb{C}\,(J\,X, F-))\,Y \to \mathbb{C}\,(J\,X, \mathrm{Lan}_J\,F\,Y)$$
$$L^F_{X,Y} =_{\mathrm{df}} [\lambda g.\, \lambda g'.\, \iota\, g \circ g']$$

Example 7. The functor $J_{\mathrm{f}} \in \mathbf{Fin} \to \mathbf{Set}$ is well-behaved. The functor $J_{\mathbf{U}} \in \mathbf{U} \to \mathbf{Cat}$ of Example 4 is well-behaved, if the type-theoretic universe $\mathsf{U} \in \mathbf{Set}, \mathsf{El} \in \mathsf{U} \to \mathbf{Set}$ is is closed under dependent products (categorically this corresponds to the induced category $\mathbf{U}$ being cartesian).

From the well-behavedness of J_{f}, it follows that $[\mathbf{Fin}, \mathbf{Set}]$ is monoidal and Lam is a monoid. These facts were proved by Fiore et al. [7].

4.2 $[\mathbb{J}, \mathbb{C}]$ Is Monoidal

Our well-behavedness conditions suffice to ensure that the unital and associativity laws of the lax monoidal structure on $[\mathbb{J}, \mathbb{C}]$ are isomorphisms. Specifically, the existence of inverses of J, K, L ensures that $\rho, \overline{\lambda}, \overline{\alpha}$ (and consequently also λ, α) have inverses too.

Theorem 6. *If* $J \in \mathbb{J} \to \mathbb{C}$ *is well-behaved, then*

1. *for any* $F \in \mathbb{J} \to \mathbb{C}$, $X \in |\mathbb{J}|$, *the map* $\rho^{-1}_{F,X} \in \mathbb{C}\,(\mathrm{Lan}_J\,F\,(J\,X), F\,X)$ *defined by* $\rho^{-1}_{F,X} =_{\mathrm{df}} [\lambda g.\, F\,(J^{-1}\,g)]$ *is an inverse of* $\rho_{F,X}$*;*
2. *for any* $X \in |\mathbb{J}|$, *the map* $\bar{\lambda}^{-1}_X \in \mathbb{C}\,(X, \mathrm{Lan}_J\,J\,X)$ *defined by* $\bar{\lambda}^{-1}_X =_{\mathrm{df}} K^{-1}\,\iota_{J,X}$ *is an inverse of* $\overline{\lambda}_X$*;*
3. *for any* $F, G \in \mathbb{J} \to \mathbb{C}$, $X \in |\mathbb{J}|$, *the map* $\bar{\alpha}^{-1}_{F,G,X} \in \mathbb{C}\,(\mathrm{Lan}_J\,F\,(\mathrm{Lan}_J\,G\,X), \mathrm{Lan}_J\,(\mathrm{Lan}_J\,F \cdot G)\,X)$ *defined by* $\bar{\alpha}^{-1}_{F,G,X} =_{\mathrm{df}} [\lambda g.\, [\lambda g'.\, \iota\, g \circ \iota\, g']\,(L^{-1}\,g)]$ *is an inverse of* $\overline{\alpha}_{F,G,X}$*.*

Hence, the category $([\mathbb{J}, \mathbb{C}], J, \cdot^J, \lambda, \rho, \alpha)$ *is monoidal.*

As an immediate corollary, we get that, in the well-behaved case, relative monads are proper monoids in a properly monoidal structure.

Corollary 1. *If* $\mathbb{J} \to \mathbb{C}$ *is well-behaved, then a relative monad* $(T, \eta, (-)^*)$ *is the same as a monoid* (T, η, μ) *in the monoidal category* $([\mathbb{J}, \mathbb{C}], J, \cdot^J, \lambda, \rho, \alpha)$*.*

4.3 Relative Monads Extend to Monads

As a pleasant bonus, the well-behavedness conditions also ensure that a relative monad extends to an ordinary monad. Crucial here is that, if J is well-behaved, we have that $\overline{\lambda}$ and $\overline{\alpha}$ are isomorphisms.

Theorem 7. *If* $J \in \mathbb{J} \to \mathbb{C}$ *is well-behaved, then a monoid* (T, η, μ) *in* $[\mathbb{J}, \mathbb{C}]$ *(equivalently, a relative monad on* J*) extends to a monoid* $(T^\#, \eta^\#, \mu^\#)$ *in* $[\mathbb{C}, \mathbb{C}]$ *(equivalently, a monad on* $\mathbb{C}$*), defined by*

$$T^\# =_{\mathrm{df}} \mathrm{Lan}_J\,T$$
$$\eta^\# =_{\mathrm{df}} I \xrightarrow{\bar{\lambda}^{-1}} \mathrm{Lan}_J\,J \xrightarrow{\mathrm{Lan}_J\,\eta} \mathrm{Lan}_J\,T$$
$$\mu^\# =_{\mathrm{df}} \mathrm{Lan}_J\,T \cdot \mathrm{Lan}_J\,T \xrightarrow{\bar{\alpha}^{-1}_{T,T}} \mathrm{Lan}_J\,(\mathrm{Lan}_J\,T \cdot T) \xrightarrow{\mathrm{Lan}_J\,\mu} \mathrm{Lan}_J\,T$$

We see that, in the well-behaved case, we can not only restrict monads to relative monads but also extend relative monads to monads. Thanks to ρ being an isomorphism, this correspondence is an embedding-projection pair.

Theorem 8. *If $J \in \mathbb{J} \to \mathbb{C}$ is well-behaved, then the correspondence between monoids in $[\mathbb{J}, \mathbb{C}]$ (equivalently, relative monads) and monoids in $[\mathbb{C}, \mathbb{C}]$ (equivalently, monads) given in Theorems 7 and 1 is an embedding-projection pair up to the natural isomorphism ρ: If (T, η, μ) is a monoid in $[\mathbb{J}, \mathbb{C}]$, then ρ_T is a monoid isomorphism between (T, η, μ) and $(T^{\#\flat}, \eta^{\#\flat}, \mu^{\#\flat})$, i.e., $\rho_T \in [\mathbb{J}, \mathbb{C}]\,(T, T^{\#\flat})$ is an isomorphism and the following diagrams in $[\mathbb{J}, \mathbb{C}]$ commute:*

$$\begin{array}{ccc} J & \xrightarrow{\eta} & T \\ & \searrow^{\eta^{\#\flat}} & \downarrow{\scriptstyle \rho_T} \\ & & T^{\#\flat} \end{array} \qquad \begin{array}{ccc} T \cdot^J T & \xrightarrow{\mu} & T \\ {\scriptstyle \rho_T \cdot^J \rho_T}\downarrow & & \downarrow{\scriptstyle \rho_T} \\ T^{\#\flat} \cdot^J T^{\#\flat} & \xrightarrow{\mu^{\#\flat}} & T^{\#\flat} \end{array}$$

In fact, $(-)^\flat$ extends to a functor from the category of monads on $\mathbb{C}$ to relative monads on J; $(-)^\#$ is its left adjoint. The relative monad maps $\rho_T \in [\mathbb{J}, \mathbb{C}]\,(T, \mathrm{Lan}_J\, T \cdot J)$ give the unit of the adjunction; the fact that it is a natural isomorphism strengthens the adjunction into a coreflection. Remarkably, this adjunction is a lifting from functors to relative monads of the adjunction $\mathrm{Lan}_J \dashv - \cdot J$ between $[\mathbb{C}, \mathbb{C}]$ and $[\mathbb{J}, \mathbb{C}]$, the defining adjunction of Lan_J.

The counit of the adjunction is $(T \cdot \overline{\lambda}) \circ \overline{\overline{\alpha}}_{T,J} \in [\mathbb{C}, \mathbb{C}]\,(\mathrm{Lan}_J\,(T \cdot J), T)$. Unlike the unit, it is generally not an isomorphism, so the adjunction is not also a reflection. For example, for $\mathbb{C} =_{\mathrm{df}} \mathbf{Set}$, $\mathbb{J} =_{\mathrm{df}} \mathbf{Fin}$, $J =_{\mathrm{df}} J_{\mathrm{f}}$, the counit is an isomorphism if and only if the monad T is finitary. This is important for us: the categories of monads on $\mathbb{C}$ and relative monads on J are generally not equivalent.

Example 8. For the powerset monad $\mathcal{P}$ on $\mathbf{Set}$, we have that $\mathcal{P}\,X$ is the powerset of a set X, $\mathcal{P}^\flat\,X =_{\mathrm{df}} \mathcal{P}\,(J_{\mathrm{f}}\,X)$ is the powerset of a finite cardinal X, and $\mathcal{P}^{\flat\#}\,X =_{\mathrm{df}} \mathrm{Lan}_{J_{\mathrm{f}}}\,\mathcal{P}^\#\,X$ is the finitary powerset (the set of finite subsets) of a (possibly infinite) set X. The difference between $\mathcal{P}$ and $\mathcal{P}^{\flat\#}$ arises because $\mathcal{P}$ is not finitary.

Example 9. For the relative monad Vec on J_{f}, $\mathsf{Vec}^\#\,X$ is the space of vectors over a possibly infinite coordinate system X that may only have finitely many non-zero components.

Example 10. For the relative monad Lam on J_{f}, we have that $\mathsf{Lam}\,X$ is the set of λ-terms over a finite, nameless context X and $\mathsf{Lam}^\#\,X$ is given by the set of λ-terms over a possibly infinite, name-carrying context X. The functor $\mathsf{Lam}^\#$ is the carrier of the initial algebra of the functor $F \in [\mathbf{Set}, \mathbf{Set}] \to [\mathbf{Set}, \mathbf{Set}]$ defined by $F\,G\,X =_{\mathrm{df}} X + G\,X \times G\,X + G\,(1 + X)$.

For the relative monad Lam^∞ the picture is different. $\mathsf{Lam}^\infty\,X$ is the set of non-wellfounded λ-terms over a finite, nameless context, but $\mathsf{Lam}^{\infty\#}\,X$ is the set of non-wellfounded λ-terms using a finite number of variables from a possibly infinite, name-carrying context. This differs from the non-finitary carrier of the final coalgebra of F, capturing general non-wellfounded λ-terms that may use infinitely many variables.

5 Arrows as a Special Case of Relative Monads

We now turn to a whole class of examples, Hughes's arrows [9]. As we shall see, arrows are relative monads on the Yoneda embedding. Arrow are commonly perceived as a generalization of monads. With relative monads, this relationship is turned upside down!

The rigorous definition of arrows by Heunen and Jacobs [8] is as follows:[1]

Definition 5. *A (***Set***-valued)* arrow *on a category* $\mathbb{J}$ *is given by*

- *a function* $R \in |\mathbb{J}| \times |\mathbb{J}| \to |\mathbf{Set}|$,
- *for any* $X, Y \in |\mathbb{J}|$, *a function* $\mathsf{pure} \in \mathbb{J}(X,Y) \to R(X,Y)$,
- *for any* $X, Y, Z \in |\mathbb{J}|$, *a function* $(\lll) \in R(Y,Z) \times R(X,Y) \to R(X,Z)$,

satisfying the conditions

- $\mathsf{pure}\,(g \circ f) = \mathsf{pure}\,g \circ \mathsf{pure}\,f$,
- $s \lll \mathsf{pure}\,\mathrm{id} = s$,
- $\mathsf{pure}\,\mathrm{id} \lll r = r$,
- $t \lll (s \lll r) = (t \lll s) \lll r$.

It follows from the conditions that R is functorial (contravariantly in the first argument), i.e., $R : \mathbb{J}^{\mathrm{op}} \times \mathbb{J} \to \mathbf{Set}$, which is the same as to say that R is an endoprofunctor on $\mathbb{J}$, and pure and $\lll$ are natural/dinatural.

A monad $(T, \eta, (-)^*)$ on $\mathbb{J}$ defines an arrow $(R, \mathsf{pure}, \lll)$ on $\mathbb{J}$ by $R(X,Y) =_{\mathrm{df}} \mathbf{Kl}(T)(X,Y)$, $\mathsf{pure}\,f =_{\mathrm{df}} L\,f$ and $\ell \lll k =_{\mathrm{df}} \ell \circ^T k$ where L is the left adjoint in the Kleisli adjunction and $\circ^T$ is the Kleisli composition.

We show now that an arrow on $\mathbb{J}$ is the same thing as a relative monad on the Yoneda embedding $\mathbf{Y} \in \mathbb{J} \to [\mathbb{J}^{\mathrm{op}}, \mathbf{Set}]$ defined by $\mathbf{Y}\,X\,Y =_{\mathrm{df}} \mathbb{J}(Y,X)$.

Theorem 9. *1. An arrow* $(R, \mathsf{pure}, \lll)$ *on* $\mathbb{J}$ *gives rise to a relative monad* $(T, \eta, (-)^*)$ *on* $\mathbf{Y}$ *defined by* $T\,X\,Y =_{\mathrm{df}} R(Y,X)$, $T\,\_\,f\,r =_{\mathrm{df}} r \lll f$, $\eta\,f =_{\mathrm{df}} \mathsf{pure}\,f$, $k^*\,r =_{\mathrm{df}} k\,\mathrm{id} \lll r$.

2. A relative monad $(T, \eta, (-)^*)$ *on* $\mathbf{Y}$ *gives rise to an arrow* $(R, \mathsf{pure}, \lll)$ *on* $\mathbb{J}$ *defined by* $R(X,Y) =_{\mathrm{df}} T\,Y\,X$, $\mathsf{pure}\,f =_{\mathrm{df}} \eta\,f$, $s \lll r =_{\mathrm{df}} (\lambda f.\, T\,\_\,f\,s)^* r$. *(The last item is well-defined, as* $\lambda f.\, T\,\_\,f\,s$ *is natural.)*

3. The above is a bijective correspondence.

The arrows on $\mathbb{J}$ and relative monads on $\mathbf{Y}$ form categories and the bijection between them extends to an equivalence of their categories.

It is easy to verify that the Freyd category of an arrow is the Kleisli category of the corresponding relative monad. Jacobs et al. [10] have previously proved that "Freyd is Kleisli for arrows" taking "Kleisli for arrows" to mean a construction that is Kleisli-like under a 2-categorical view of the Kleisli construction for monads. We can take it to mean "Kleisli for arrows as relative monads".

[1] Since we compare arrows to monads, not strong monads, we mean "weak" arrows here: $\mathbb{J}$ does not have to be symmetric monoidal and no first operation is required.

The Yoneda embedding is well-behaved. We reconstruct the result of Heunen and Jacobs [8] about arrows being monoids as an instance of a generality.

Theorem 10. *If $\mathbb{J}$ is small, then* $\mathbf{Y}$ *is well-behaved, hence the category $[\mathbb{J}, [\mathbb{J}^{\mathrm{op}}, \mathbf{Set}]]$ is monoidal. An arrow on $\mathbb{J}$ is a monoid in this category.*

Jacobs and Heunen considered the special case of arrows and showed an arrow to be a monoid in $[\mathbb{J}^{\mathrm{op}} \times \mathbb{J}, \mathbf{Set}]$ (the category of endoprofunctors on $\mathbb{J}$) as a monoidal category, which is an equivalent statement.

6 Conclusions and Further Work

We have introduced a generalisation of monads, relative monads, which is motivated by examples and subsumes arrows, a well-known generalisation of monads. Indeed, when moving to a more precise type discipline, the illusion that everything takes place in only one ambient category (say, **Set**) can no longer be maintained and as a consequence we have to revisit the categorically inspired concepts of functional programming. We believe that our examples demonstrate that monad-like entities which are not endofunctors are natural; fortunately, they are precisely monoids in the functor category. We also suggest that our presentation of relative monads given in Sect. 2.1 is accessible for functional programmers, indeed it does not differ substantially from ordinary monads.

Our development is only the first step. Due to lack of space, we have not written about monad maps; we did not comment on the relationship between relative adjunctions and adjunctions etc.; strong monads (esp. versus strong arrows) are a further additional topic. We will elsewhere comment on the relation of our relative monads to the the recent generalization of monads by Spivey [17] that was also motivated by programming examples: he fixes a functor $K \in \mathbb{C} \to \mathbb{J}$ (notice the direction) to then look for monad-like structures with an underlying functor $\mathbb{J} \to \mathbb{C}$. With Paul Levy we have checked that a fair amount of monad theory transfers to his generalized monads, but they are not monoids in $[\mathbb{J}, \mathbb{C}]$ unless K has a left adjoint, in which case they are equivalent to relative monads.

It seems clear that many of the concepts known from ordinary monads carry over to the relative setting. We have already mentioned Jaskelioff's work on monad transformers which is expressed in a general monoidal setting and hence carries over to relative monads. We hope that this generalisation of the monadic approach leads to new programming structures supporting a greater reusability of concepts and programs.

Acknowledgements. We are grateful to Paul Levy for valuable comments and hints. T. Altenkirch was supported by the Engineering and Physical Sciences Research Council (EPSRC) grant no. EP/G034109/1. J. Chapman and T. Uustalu were supported by the Estonian Centre of Excellence in Computer Science, EXCS, financed by the European Regional Development Fund. T. Uustalu was also supported by the Estonian Science Foundation grant no. 6940.

References

1. Abbott, M., Altenkirch, T., Ghani, N.: Containers—constructing strictly positive types. Theor. Comput. Sci. 342(1), 3–27 (2005)
2. Abramsky, S., Coecke, B.: A categorical semantics of quantum protocols. In: Proc. of 19th Ann. IEEE Symp. on Logic in Computer Science, LICS 2004, pp. 415–425. IEEE CS Press, Los Alamitos (2004)
3. Agda team: Agda (2009), `http://appserv.cs.chalmers.se/users/ulfn/wiki/agda.php`
4. Altenkirch, T., Green, A.: The Quantum IO Monad. In: Gay, S., McKie, I. (eds.) Semantic Techniques in Quantum Computation, pp. 173–205. Cambridge Univ. Press, Cambridge (2009)
5. Altenkirch, T., Reus, B.: Monadic presentations of lambda terms using generalized inductive types. In: Flum, J., Rodríguez-Artalejo, M. (eds.) CSL 1999. LNCS, vol. 1683, pp. 453–468. Springer, Heidelberg (1999)
6. Fiore, M.: Semantic analysis of normalisation by evaluation for typed lambda calculus. In: Proc. of 4th ACM SIGPLAN Conf. on Principles and Practice of Declarative Programming, PPDP 2002, pp. 26–37. ACM Press, New York (2002)
7. Fiore, M., Plotkin, G., Turi, D.: Abstract syntax and variable binding. In: Proc. of 14th Ann. IEEE Symp. on Logic in Computer Science, LICS 1999, pp. 193–202. IEEE CS Press, Los Alamitos (1999)
8. Heunen, C., Jacobs, B.: Arrows, like monads, are monoids. In: Brookes, S., Mislove, M. (eds.) Proc. of 22nd Ann. Conf. on Mathematical Foundations of Programming Semantics, MFPS XXII. Electron. Notes in Theor. Comput. Sci, vol. 158, pp. 219–236. Elsevier, Amsterdam (2006)
9. Hughes, J.: Generalising monads to arrows. Sci. of Comput. Program. 37(1-3), 67–111 (2000)
10. Jacobs, B., Heunen, C., Hasuo, I.: Categorical semantics for arrows. J. of Funct. Program. 19(3-4), 403–438 (2009)
11. Jaskelioff, M.: Lifting of Operations in Modular Monadic Semantics. PhD thesis, University of Nottingham (2009)
12. Manes, E.G.: Algebraic Theories. Springer, Heidelberg (1976)
13. McBride, C., Paterson, R.: Applicative programming with effects. J. of Funct. Program. 18(1), 1–13 (2008)
14. Morris, P., Altenkirch, T.: Indexed containers. In: Proc. of 24th Ann. IEEE Symp. on Logic in Computer Science, LICS 2009, pp. 277–285. IEEE CS Press, Los Alamitos (2009)
15. Piponi, D.: Commutative monads, diagrams and knots. In: Proc. of 14th Int. Conf. on Functinal Programming, ICFP 2009, p. 231. ACM Press, New York (2009) (see the video)
16. Power, J., Robinson, E.: Premonoidal categories and notions of computation. Math. Struct. in Comput. Sci. 7(5), 453–468 (1997)
17. Spivey, J.M.: Algebras for combinatorial search. J. of Funct. Program. 19(3-4), 469–487 (2009)
18. Uustalu, T., Vene, V.: Comonadic notions of computation. In: Adamék, J., Kupke, C. (eds.) Proc. of 9th Int. Wksh. on Coalgebraic Methods in Computer Science, CMCS 2008. Electron. Notes in Theor. Comput. Sci., vol. 203(5), pp. 263–284. Elsevier, Amsterdam (2008)
19. Vizzotto, J.K., Altenkirch, T., Sabry, A.: Structuring quantum effects: Superoperators as arrows. Math. Struct. in Comput. Sci. 16(3), 453–468 (2006)