A Kleisli-style construction for Parametric Monads, with Examples

University of Bath, Claverton Down Road, Bath. BA2 7AY,
wjg27@bath.ac.uk,
WWW home page: http://people.bath.ac.uk/wjg27

Abstract. A well-known principle in denotational semantics is that adding an effect to a categorical model should correspond to the Kleisli category construction for some suitable monad. Our point of departure is the observation that much recent work in semantics, particularly in game semantics, has broken away from this principle by modelling effects using categories that are not Kleisli categories. Our aim is to extend the theory of monadic effects to these models in a systematic way.
An important part of our approach will be to study parametric monads, otherwise known as lax actions of monoidal categories. These generalize monads: a monad is precisely an action of the trivial category. We introduce a construction on parametric monads which takes a parametric monad on a category and gives us back a new category. As with the Kleisli category, the objects of this category will be the objects of the original category, while the morphisms are obtained from morphisms in the original category in a formal manner brought about by the action. In the case that the action is actually a monad, we get the usual Kleisli category.
We show that this new construction is closely related to existing models of effects in game semantics, allowing us to reason systematically about adequacy and full abstraction proofs. We illustrate the relevance of the construction by showing that a special case gives us a fully abstract game semantics for Probabilistic Algol, which can be related to the existing model given by Danos and Harmer.

Keywords: denotational semantics, category theory, game semantics

1 Introduction

1.1 Monads and Kleisli categories

This paper is about general frameworks for modelling effects in a categorical semantics. The seminal paper in this field is Moggi's *Computational Lambda-Calculus and Monads* [1], which first put forward the idea that if $\mathcal{C}$ is a categorical model of a programming language, then we can simulate an effect in that language via a suitable *monad* on $\mathcal{C}$. Given such a monad, its Kleisli category $Kl_M\mathcal{C}$ is then a model of a language formed by adding the effect to the original language.

Spivey [2] and Wadler [3, 4] have transferred this theory to the real world by using monads as programming tools, particularly in the Haskell programming language.

Monads are certainly not the only formal method of adding effects into categories. For example, [5] argues that Lawvere theories, a tool used originally in universal algebra (as monads were), also give us a natural way to model effects in categories. Since [6] every Lawvere theory may be regarded as a particular monad on the category of sets, this is not surprising. However, Lawvere theories are a lot more flexible than monads when it comes to composing multiple effects together.

More recent work in game semantics, however, has produced models of effects – such as the well-known game semantics for scoped state [7] and nondeterminism [8] – do not arise by applying any systematic procedure to a base category of games that lacks the effect. Instead, these models are typically built up using intuition – for example, by relaxing the determinism condition on strategies to give a model of a nondeterministic language – and do not readily fit into any category theoretic framework the way Kleisli categories do. The goal of this paper is to examine some possible frameworks that will allow us to study these and other models in a systematic way.

A closely related problem is that of using a common technique to model a particular effect in different settings. Many of our favourite monads (e.g., the powerset monad on **Set** for nondeterminism) live in some particular category and cannot readily be set up in other categories.

Before we start, it is worth mentioning one example of a Kleisli category arising naturally in game semantics. The *slot games* defined in [9] are defined like ordinary games, but their strategies are allowed to contain extra moves, called *tokens*, which are meant to represent a point at which the computation takes up time or resources. For example, a strategy for a term of natural number type that returns the value 5 after two CPU cycles could be represented by the strategy shown on the left of Figure 1.1. Now, we can alternatively represent this strategy as being a normal strategy inside the Kleisli category for the reader monad given by the *command* or *singleton* type, as on the right of Figure 1.1. The translation works by replacing each token (\$) with a request to the argument to the function: since this arguement has singleton type, there is only one possible value that can be passed in, but the game semantics will record that we have made the request. What is more, this translation respects composition in the two categories, both the Kleisli composition and the composition defined synthetically for slot games.

Since the reader monad is entirely built out of the Cartesian closed structure of the category, we can perform a similar trick in many different settings, although perhaps not as successfully as in game semantics.

1.2 Promonads

One clue comes from generalizing from functors to profunctors. Since a monad is a monoid in the category End[$\mathcal{C}$] of endofunctors on $\mathcal{C}$ with monoidal product

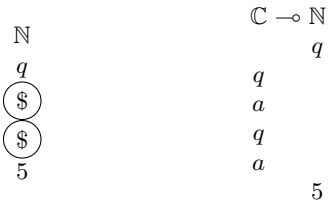


Fig. 1. The 'slot-games' denotation of a program that uses resources can also be represented in a Kleisli category.

given by functor composition, we can define a *promonad* to be a monoid in the category $\mathrm{Prof}[\mathcal{C},\mathcal{C}]$ of endoprofunctors on $\mathcal{C}$ (i.e., functors $\mathcal{C}^{op}\times\mathcal{C}\to\mathbf{Set}$), where the monoidal product is profunctor composition. Since a functor $\mathcal{F}\colon\mathcal{C}\to\mathcal{D}$ may be identified with the profunctor $\mathcal{D}[F(\ \_\ ),\ \_\ ]\colon\mathcal{C}^{op}\times\mathcal{D}\to\mathbf{Set}$, this certainly generalizes the idea of a monad.

A little thought, however, shows that a promonad is a vast generalization of a monad. Any promonad $\mathcal{D}\colon\mathcal{C}^{op}\times\mathcal{C}\to\mathbf{Set}$ on a category $\mathcal{C}$ may be identified [**?**] with a category (also called $\mathcal{D}$) where the objects are the objects of $\mathcal{C}$ and the morphisms from A to B are elements of the set $\mathcal{D}(A,B)$. The promonad structure gives us composition and identities in the category $\mathcal{D}$, and also a functor $J\colon\mathcal{C}\to\mathcal{D}$ that is the identity on objects.

In the other direction, if we have such a $\mathcal{C}\xrightarrow{J}\mathcal{D}$, then the hom functor $\mathcal{D}(J(\ \_\ ),J(\ \_\ )$ is a promonad on $\mathcal{C}$. Thus, promonads on $\mathcal{C}$ generalize the Kleisli construction to such an extent that they incorporate, among others, any category formed from $\mathcal{C}$ by keeping the same objects and adding new morphisms, such as the game semantics models of state [7] and nondeterminism [8].

On the other hand, what promonads capture is quite a natural simple way of modelling effects: keep the same types (objects), but give us new ways to write programs (morphisms). Although effectful types are important (e.g., in work on call-by-push-value [**?**]), we shall adopt this paradigm in our paper. So one way to state our goal is that we are looking for systematic ways to construct promonads on a category that are more general than the existing Kleisli construction for monads.

1.3 Parametric monads

Melliès [10], following work by Smirnov [11], has demonstrated that the concept of a lax monoidal category action is a useful generalization of that of a monad. A lax action of a monoidal category $\mathcal{M}$ upon a category $\mathcal{C}$ is a lax monoidal functor $\mathcal{M}\to\mathrm{End}[\mathcal{C}]$ (considered as a functor $\ \_\cdot\_\ :\mathcal{M}\times\mathcal{C}\to\mathcal{C}$); monads are the special case when $\mathcal{M}$ is trivial. For this reason, Melliès renames lax monoidal actions to 'parametric monads', where we think of the action of $\mathcal{M}$ as a kind of monad on $\mathcal{C}$ that is parameterized by the objects of $\mathcal{M}$.

This idea has been further generalized by Katsumata [12], who has given a definition of a 'Kleisli-like resolution' of a parametric monad. This particular construction, which takes a parametric monad on a category $\mathcal{C}$ parameterized by a monoidal category $\mathcal{M}$ and yields a new category, generalizes many important properties of the Kleisli category – in particular, it gives us an adjunction with the original category – but it does not fit into our framework, since the objects of the category it creates are not the objects of the original category $\mathcal{C}$, but pairs of an object of $\mathcal{C}$ and an object of $\mathcal{M}$.

Once again, it is useful to generalize to profunctors. First, if $\mathcal{C}$ is a category and $\mathcal{M}$ a monoidal category, we define a *parametric promonad on $\mathcal{C}$ parameterized by $\mathcal{M}$* to be a lax monoidal functor $\mathcal{M} \to \mathrm{Prof}[\mathcal{C}, \mathcal{C}]$. In a similar way to before, if we have a parametric monad $\_ \,.\, \_ : \mathcal{M} \times \mathcal{C} \to \mathcal{C}$ then we may identify it with the parametric promonad given by $\mathcal{D}(X, A, B) = \mathcal{C}(A, X.B)$.

As with promonads, we can now take a parametric promonad $\mathcal{D} \colon \mathcal{M} \times \mathcal{C}^{op} \times \mathcal{C} \to \mathbf{Set}$ and try to recast it as a particular kind of category. There are various ways to do it, but one is to view it as a bicategory whose objects are the objects of $\mathcal{C}$, where the elements of the set $\mathcal{D}(X, A, B)$ are considered as 1-morphisms from A to B and where morphisms $X \to Y$ in $\mathcal{M}$ give rise to 2-morphisms between the elements of $\mathcal{D}(X, A, B)$ and the elements of $\mathcal{D}(Y, A, B)$.

A natural idea is to turn this bicategory into a 2-category by quotienting out by the action of the 2-morphisms, which gives us the following collapse from parametric promonads to promonads.

$$\mathcal{D} \mapsto \varinjlim_{X \colon \mathcal{M}} \mathcal{D}(X, \_, \_)$$

In the particular case that the parametric promonad is in fact a parametric *monad* (so $\mathcal{D}(X, A, B) = \mathcal{C}(A, X.B)$), then we will call the category we obtain $\mathcal{C}/\mathcal{M}$. The morphisms in $\mathcal{C}/\mathcal{M}$ are given by

$$\mathcal{C}/\mathcal{M}(A, B) = \varinjlim_{X \colon \mathcal{M}} \mathcal{C}(A, X.B)\,,$$

with the composition of $f \colon A \to X.B$ and $g \colon B \to Y.C$ being given by the composite

$$A \xrightarrow{f} X.B \xrightarrow{X.g} X.Y.C \to (X \otimes Y).C\,.$$

Note that if $\mathcal{M}$ is the trivial category (so that the action is in fact a monad), then the category $\mathcal{C}/\mathcal{M}$ is the usual Kleisli category.

1.4 Effects, monads and colimits

The colimit that appears in the definition of the category $\mathcal{C}/\mathcal{M}$ is instructive, because it is linked to the fact that many important monads may be expressed as colimits. For example, let $\mathbf{Surj}^+$ be the category of non-empty sets and sur-

jections. Then, if A is a set, we have

$$\begin{aligned} \varinjlim_{X\colon \mathbf{Surj}^{+\,op}} [X, A] &\cong \mathcal{P}(A) \\ \left(X \xrightarrow{f} A\right) &\mapsto \mathrm{Im}(f)\,, \end{aligned}$$

naturally in A.

This is useful, because the body of the colimit – i.e., the function set $[X, A]$ – has a natural equivalent inside any Cartesian closed category $\mathcal{C}$ that admits a functor $\mathcal{F}\colon \mathbf{Set} \to \mathcal{C}$; i.e., the internal hom $\mathcal{F}(X) \multimap A$. Moreover, the functor $[X, A]$ is actually a lax action of $\mathbf{Surj}^{+,op}$ (with the Cartesian product) upon the category of sets, and the monad structure of the powerset may be deduced from the structure of this action via the colimit. So even though we cannot define the powerset *monad* inside arbitrary Cartesian closed categories, we can still define this action of $\mathbf{Surj}^{op}$.

The idea, then, is that we might try to model nondeterminism in other categories (where we might not be able to define the powerset) using this parametric monad. One unfortunate fact is that these two alternatives (powerset monad vs lax action of $\mathbf{Surj}^{+,op}$) do not give rise to the same promonad in the category of sets. Indeed, if we have a general action of a monoidal category $\mathcal{M}$ on $\mathbf{Set}$, then (as long as the possibly large colimits exist) we get a monad on $\mathbf{Set}$ given by

$$MA = \varinjlim_{X\colon \mathcal{M}} X.A$$

and the morphisms between sets A and B in the Kleisli category are given by

$$[A, \varinjlim_{X\colon \mathcal{M}} X.B]\,.$$

Meanwhile, the morphisms between A and B in the category $\mathcal{C}/\mathcal{M}$ are given by the set

$$\varinjlim_{X\colon \mathcal{M}} [A, X.B]\,,$$

and the two sets need not be the same, since the colimit might not commute with the functor $[A,\ \_\,]$.

However, if the category $\mathcal{M}$ satisfies certain additional properties, we can get some relation back. There is always a natural identity-on-objects functor between the two categories, given by the natural map

$$\varinjlim_{X\colon \mathcal{M}} [A, X.B] \to [A, \varinjlim_{X\colon \mathcal{M}} X.B]\,,$$

and if the category $\mathcal{M}$ admits a cocone over every discrete diagram (i.e., if whenever (X_i) is a collection of objects of $\mathcal{M}$ there is some Y such that there is a morphism $f_i\colon X_i \to Y$ for all i), then this function is surjective, meaning that the functor will be full. The category $\mathbf{Surj}^{+,op}$ has this property (via the

Cartesian product and its projections, which are always surjective for non-empty sets). In the case of the action of $\mathbf{Surj}^{+,op}$ that gives rise to the powerset monad, the natural map given above is the surjection $\mathcal{P}[A,B] \to [A,\mathcal{P}B]$ that takes a set $\mathcal{F}$ of functions $A \to B$ and forms the function $A \to \mathcal{P}B$ that sends an element a to the set

$$\{f(a) : f \in \mathcal{F}\}.$$

This means that we may recover the Kleisli category from the category given by the action by taking the quotient by a further equivalence relation on morphisms.

In general, if a monad M on **Set** can be expressed as the colimit of a lax action of a monoidal category $\mathcal{M}$ on **Set**, where $\mathcal{M}$ admits a cocone over every discrete diagram, then we do not lose very much by considering the category $\mathbf{Set}/\mathcal{M}$ obtained from the action rather than the Kleisli category $\mathrm{Kl}_M\,\mathbf{Set}$. More specifically, there is a full functor $\mathrm{Kl}_M\,\mathbf{Set} \to \mathbf{Set}/\mathcal{M}$ that is the identity on objects, so we may recover the Kleisli category by applying a suitable equivalence relation to the set of morphisms. The benefit of reasoning about the action of $\mathcal{M}$ rather than the monad M is that the action might be realizable in a much larger class of categories. For instance, the action of $\mathbf{Surj}^{+\,op}$ on **Set** may be replicated inside any monoidal closed category that admits a functor out of the category of sets (for example, most models of PCF or Idealized Algol). We will see later that the non-empty powerset monad has a probabilistic counterpart, where the set of discrete probability distributions on a set may be obtained as a colimit over the category of probability spaces and probability-preserving maps, which satisfies similar properties, allowing us to model discrete probability in categories other than **Set**.

1.5 Plan for the paper

The rest of this paper will consist in two sections. In the first, we will give a more detailed account of the construction that allows us to build a promonad (i.e., a category) out of a parametric monad.

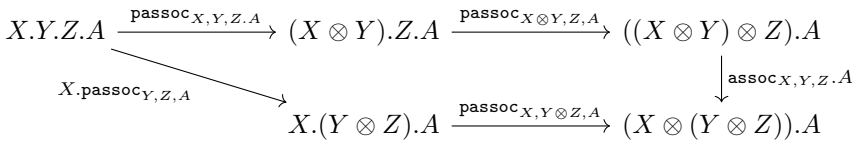
In the second section, we will give a fully abstract game semantics for the language Probabilistic Algol (PA). This is by no means a new result – the paper [14] that introduced PA also gave it a fully abstract game semantics. Instead, the point of this section will be to demonstrate how the ideas we have outlined can be applied systematically to yield fully abstract models of programming languages.

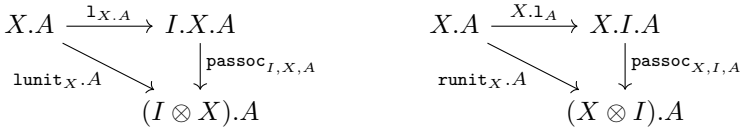
2 A Kleisli-style construction for parametric monads

2.1 Parametric monads

Let $\mathcal{M}$ be a monoidal category and let $\mathcal{C}$ be a category. Then a *lax left action* of $\mathcal{M}$ on $\mathcal{C}$ is a functor $\_ \,.\, \_ : \mathcal{M} \times \mathcal{C} \to \mathcal{C}$ that gives rise (through currying) to a lax monoidal functor $\mathcal{M} \to \mathrm{End}[\mathcal{C},\mathcal{C}]$. In other words, we have natural

transformations $\mathtt{passoc}_{X,Y,A}\colon X.Y.A \to (X \otimes Y).A$ and $\mathtt{l}_A\colon A \to I.A$ making the following diagrams commute for all objects A of $\mathcal{C}$ and X, Y, Z of $\mathcal{M}$.





Example 1. – Any monad M is an action of the trivial category on its underlying category $\mathcal{C}$, regarding MA as the action $I.A$. The natural transformation $\mathtt{passoc}_{A,I,I}$ is precisely the monad action on A:

$$I.I.A = MMA \to MA = I.A = (I \otimes I).A\,.$$

Dually, any lax action gives rise to a monad on the category being acted upon, by setting $MA = A.I$.

– If $\mathcal{C}$ is also a monoidal category and $J\colon \mathcal{M} \to \mathcal{C}$ is a lax monoidal functor with monoidal coherence μ, then there is an action of $\mathcal{M}^{co}$ (i.e., $\mathcal{M}$ with the opposite monoidal product) on $\mathcal{C}$ given by

$$A.X = A \otimes JX\,,$$

and

$$\mathtt{passoc}_{X,Y,A} = (A{\otimes}JY){\otimes}JX \xrightarrow{\mathtt{assoc}_{A,JX,JY}} A{\otimes}(JY{\otimes}JX) \xrightarrow{A\otimes\mu_{X,Y}} A{\otimes}J(Y{\otimes}X)\,.$$

We sometimes refer to a left action of the opposite category $\mathcal{M}^{co}$ on $\mathcal{C}$ as a *right action* of $\mathcal{M}$ on $\mathcal{C}$. In that case, we write $X.A$ instead of $A.X$, so that the coherences become

$$\mathtt{passoc}_{A,X,Y}\colon A.X.Y \to A.(X \otimes Y)$$
$$\mathtt{r}_A\colon A \to A.I\,.$$

– If $\mathcal{C}$ is a monoidal closed category, and j an oplax monoidal functor with coherence ν, then there is an action of $\mathcal{M}^{op}$ on $\mathcal{C}$ given by

$$A.X = jX \multimap A$$

and

$$\mathtt{passoc}_{A,X,Y} = jY \multimap (jX \multimap A) \to (jY{\otimes}jX) \multimap A \xrightarrow{\nu_{Y,X}\multimap A} j(Y{\otimes}X) \multimap A\,.$$

- The intersection of the first two examples is the *writer monad* given by $M_W X = X \otimes W$ for any monoid W in $\mathcal{C}$. The intersection of the first and third examples is the *reader monad* given by $M^R X = R \multimap X$ for any comonoid R in $\mathcal{C}$ (and, in particular, for any object R if $\mathcal{C}$ if the monoidal product in $\mathcal{C}$ is Cartesian).
  We shall therefore call an action of the form $A \otimes JX$ a *writer-style action* and one of the form $jX \multimap A$ a *reader-style action*.
- We can define an *oplax action* of $\mathcal{M}$ on $\mathcal{C}$ to be a lax action of $\mathcal{M}$ upon the opposite category $\mathcal{C}^{op}$. In this case, the coherence `passoc` goes from $A.(X \otimes Y)$ to $A.X.Y$. As monads are lax actions of the trivial category, so are comonads oplax actions of the trivial category.
  Since the functors $\_ \otimes U$ and $U \multimap \_$ are adjoint, any reader-style lax action $jX \multimap A$ may also be regarded as an oplax action $A \otimes jX$. We shall refer to these oplax actions as *reader-style* too.

2.2 The $\mathcal{C}/\mathcal{M}$ construction

Suppose we have a lax action of a monoidal category $\mathcal{M}$ upon a category $\mathcal{C}$. We define a new category $\mathcal{C}/\mathcal{M}$ as follows. The objects of $\mathcal{C}/\mathcal{M}$ are the objects of $\mathcal{C}$, while the morphisms are given by the following formula.

$$\mathcal{C}/\mathcal{M}(A, B) = \varinjlim_{X \colon \mathcal{M}} \mathcal{C}(A, X.B)$$

That is, if A and B are objects of $\mathcal{C}$, then a morphism from A to B in $\mathcal{C}/\mathcal{M}$ is an equivalence class of pairs (X, f), where X is an object of $\mathcal{M}$ and $f \colon A \to X.B$ a morphism in $\mathcal{C}$, and where (X, f) and (Y, g) are said to be equivalent if there is a morphism $h \colon X \to Y$ in $\mathcal{M}$ making the following diagram commute.

$$\begin{array}{ccc} A & \xrightarrow{f} & X.B \\ & \searrow^{g} & \downarrow{\scriptstyle h.B} \\ & & Y.B \end{array}$$

Note that this does not automatically define an equivalence relation in general, so there may be pairs (X, f) and (Y, g) that are not related by some $h \colon X \to Y$ but are nevertheless equivalent because there is some chain of such relations from one to the other.

For the rest of the paper, we will refer to the pair (X, f) using the morphism f, since the object X should usually be clear from context.

Let A, B, C be objects of $\mathcal{C}$, and let $f \colon A \to X.B, g \colon B \to Y.C$ be morphisms $A \to B$ and $B \to C$ in $\mathcal{C}/\mathcal{M}$. We define the composition of f with g to be given by the following composite in $\mathcal{C}$.

$$A \xrightarrow{f} X.B \xrightarrow{X.g} X.Y.C \xrightarrow{\mathtt{passoc}_{X,Y,C}} (X \otimes Y).C$$

The identity morphism $A \to A$ is given by the morphism $\mathtt{1}_A \colon A \to I.A$ in $\mathcal{C}$. The coherence conditions for the action guarantee that this is an identity and that our composition is associative.

If we consider the case that $\mathcal{M}$ is trivial, so the action is a monad M given by $MA = I.A$, then this composition becomes

$$A \xrightarrow{f} MB \xrightarrow{Mg} MMC \xrightarrow{\mathtt{passoc}_{I,I,C}} MC\,,$$

which is precisely the usual Kleisli composition.

There is a natural functor $J\colon \mathcal{C} \to \mathcal{C}/\mathcal{M}$ that is the identity on objects and that sends the morphism $f\colon A \to B$ in $\mathcal{C}$ to the composite

$$A \xrightarrow{f} B \xrightarrow{1_B} I.B\,,$$

considered as a morphism $A \to B$ in $\mathcal{C}/\mathcal{M}$.

Let us see what that looks like when our action is the action of $\mathbf{Surj}^{+\,op}$ on $\mathbf{Set}$ that we considered earlier. In that case, a morphism $A \to B$ is given by a function $A \to [X, B]$, or equivalently $X \to [A, B]$. We may identify this function with its image, which is a subset of $[A, B]$, and this identification corresponds precisely to the equivalence relation that is automatically given by our construction; namely, the smallest equivalence relation such that $X \to [A, B]$ is equivalent to the composite $Y \to X \to [A, B]$ for any surjection $Y \to X$. Indeed, if $f\colon X \to [A, B]$ and $g\colon Y \to [A, B]$ have the same image $\mathcal{F} \subset [A, B]$, then f and g are both equivalent to the inclusion $\mathcal{F} \hookrightarrow [A, B]$, so they are equivalent.

2.3 Universal property of $\mathcal{C}/\mathcal{M}$

The category $\mathcal{C}/\mathcal{M}$ has one other piece of structure besides the functor out of $\mathcal{C}$. If X is an object of M and A and object of $\mathcal{C}$, then the identity morphism $X.A \to X.A$ may be considered as a morphism $\phi_{X,A}\colon J(X.A) \to JA$ in $\mathcal{C}/\mathcal{M}$. $\phi_{X,A}$ is natural in X and A and also makes the following diagram commute.

$$\begin{array}{ccc} J(X.Y.A) & \xrightarrow{\phi_{X,Y.A}} & J(Y.A) \\ {\scriptstyle J\mathtt{passoc}_{X,Y,A}}\downarrow & & \downarrow{\scriptstyle \phi_{Y,A}} \\ J((X \otimes Y).A) & \xrightarrow{\phi_{X\otimes Y,A}} & JA \end{array}$$

We call a natural transformation $\psi_{X,A}\colon J(X.A) \to JA$ *multiplicative* if it satisfies the same commutative diagram.

$\mathcal{C}/\mathcal{M}$, J and $\phi_{X,A}$ are universal in the following sense.

Proposition 1. *i) Let $\mathcal{C}/\mathcal{M}$, J and $\phi_{X,A}$ be as above. Let $\mathcal{D}$ be a category, F a functor $\mathcal{C} \to \mathcal{D}$ and $\psi_{X,A}\colon F(X.A) \to FA$ a multiplicative natural transformation in $\mathcal{D}$. Then there is a unique functor $H\colon \mathcal{C}/\mathcal{M} \to \mathcal{D}$ such that $F = HJ$ and $\psi = H\phi$.*

ii) Let $\mathcal{D}$ be a category and let $H, K\colon \mathcal{C}/\mathcal{M} \to \mathcal{D}$ be two functors. Let $\alpha\colon HJ \to KJ$ be a natural transformation making the following diagram commute.

$$\begin{array}{ccc} HJA & \xrightarrow{\alpha_A} & KJA \\ {\scriptstyle H\phi_{X,A}}\downarrow & & \downarrow{\scriptstyle K\phi_{X,A}} \\ HJX.A & \xrightarrow{\alpha_{X.A}} & KJA \end{array}$$

Then there is a unique natural transformation $\beta\colon H \to K$ *such that* $\alpha = \beta J$.

Proof. The proof, which we will not give full details of, comes down to the fact that we can factorize any morphism $f\colon A \to X.B$ from A to B in $\mathcal{C}/\mathcal{M}$ as:

$$f = A \xrightarrow{Jf} X.B \xrightarrow{\phi_{X,A}} B\,,$$

where we have considered f both as a morphism $A \to B$ in $\mathcal{C}/\mathcal{M}$ and as a morphism $A \to X.B$ in $\mathcal{C}$. It therefore suffices to define H on morphisms of the form Jf (which must be sent to Ff, since $F = HJ$) and on morphisms of the form $\phi_{X,A}$ (which must be sent to $\psi_{X,A}$, since $\psi = H\phi$). Therefore H is uniquely defined, if it exists, so it suffices to check that the H given by sending the above factorization to the composite

$$FA \xrightarrow{Ff} F(X.B) \xrightarrow{\psi_{X,A}} FB$$

is indeed a functor satisfying the requirements of the proposition. Part (ii) is proved in a very similar way.

Remark 1. If $\mathcal{M}$ is the trivial category, so our action is a monad M, then this universal property can be used to prove the usual universal property for the Kleisli category; namely, that it is initial among all adjunctions giving rise to M. To see this, suppose that $\mathcal{C} \xrightarrow{F} \mathcal{D} \xrightarrow{G} \mathcal{C}$ are functors such that F is left adjoint to G and $M = GF$. Then we have a natural transformation

$$FMA = FGFA \xrightarrow{\epsilon_{FA}} FA$$

which satisfies the hypotheses of Proposition 2(i). Therefore, Proposition 2 tells us that the Kleisli category is not only initial amongst adjunctions giving rise to M, but is initial among all functors $F\colon \mathcal{C} \to \mathcal{D}$ that admit a natural transformation $FMA \to FA$. Note that for more general $\mathcal{M}$, we do not in general get an adjunction between $\mathcal{C}$ and $\mathcal{C}/\mathcal{M}$.

The quotient notation we have used is because the universal property we have outlined exhibits $\mathcal{C}/\mathcal{M}$ as a particular weighted coequalizer of the functors pr_2, $\_ \,.\, \_ : \mathcal{M} \times \mathcal{C} \rightrightarrows \mathcal{C}$ in **Cat**. It may therefore be regarded as a lax 2-dimensional analogue of the quotient X/G of a set X by a group or monoid action.

When the action is a monad, this is a strange idea, since we are not used to thinking of, for example, a Kleisli category as quotient of the original category. The reason for this is the laxness: we do not identify the objects A and $X.A$ with an isomorphism, but with a morphism in one direction. One way in which a Kleisli category does act like a quotient is that it allows us to transfer effects that are usually associated with higher types on to lower types. For example, the type $W \to [A, W]$ encodes a state transformer with state W. Working in the Kleisli category for the state monad allows us to associate this stateful behaviour to the object A.

Remark 2. The proof of Proposition 2 tells us that one way of thinking of the category $\mathcal{C}/\mathcal{M}$ is as a general theory of categories that extend $\mathcal{C}$, and admit a factorization result whereby every morphism is the composition of a morphism from $\mathcal{C}$ with one of the distinguished morphisms $\phi_{X,A}$.

Factorization results are very important in the game semantics of effects. What usually happens is that when we extend a model $\mathcal{G}$ to a new model $\mathcal{G}'$, we wish to show that each morphism in $\mathcal{G}'$ may be written as the composite of a morphism from $\mathcal{G}$ composed with one of a small collection of new morphisms that are readily definable in the language – for example, nondeterministic oracles, or simple storage cells. This allows us to lift a definability result from the old category to the new one. If these new morphisms can be cast as the components of some natural transformation $\psi_{X,A} \colon X.A \to A$ for some action of some monoidal category $\mathcal{M}$ on $\mathcal{G}$ (and often they can), then Proposition 2(i) gives us a functor from $\mathcal{G}/\mathcal{M}$ to $\mathcal{G}'$. The factorization result then tells us that this functor is full, and so the model $\mathcal{G}'$ can also be defined as a quotient of $\mathcal{G}/\mathcal{M}$.

2.4 Monoidal and monoidal closed structure of $\mathcal{C}/\mathcal{M}$

We saw in the last section that the $\mathcal{C}/\mathcal{M}$ construction may be regarded as a kind of lax 2-coequalizer. In the category of sets, if a pair of morphisms $f, g \colon A \rightrightarrows B$ is *reflexive* – i.e., if there is a morphism $h \colon B \to A$ such that $f \circ h = g \circ h = \mathrm{id}_B$, then the coequalizer of f and g commutes with finite products.

In our setting, the two functors pr_2 and $\_ \,.\, \_$ from $\mathcal{M} \times \mathcal{C} \to \mathcal{C}$ form a reflexive pair, since they have a common section $h \colon \mathcal{C} \to \mathcal{M} \times \mathcal{C}$ given by $h(A) = (I, A)$. It turns out that a similar result then applies.

Proposition 2. *Given an action of a monoidal category $\mathcal{M}$ on a category $\mathcal{C}$, and an action of a monoidal category $\mathcal{M}'$ on a category $\mathcal{C}'$, there is an obvious way to define an action of $\mathcal{M} \times \mathcal{M}'$ on $\mathcal{C} \times \mathcal{C}'$. Then the functor*

$$(\mathcal{C} \times \mathcal{C}')/(\mathcal{M} \times \mathcal{M}') \to (\mathcal{C}/\mathcal{M}) \times (\mathcal{C}'/\mathcal{M}')$$

induced as in Proposition 2 is an isomorphism.

Now suppose that $\mathcal{C}$ is itself a monoidal category. We say that the action of $\mathcal{M}$ on $\mathcal{C}$ is *monoidal* if it is a lax monoidal functor $\mathcal{M} \times \mathcal{C} \to \mathcal{C}$, which means that we need to have a natural transformation

$$X.A \otimes Y.B \to (X \otimes Y).(A \otimes B)\,,$$

together with appropriate coherences.

If the action is reader-style, given by $X.A = jX \multimap A$ for an oplax monoidal functor $j \colon \mathcal{M} \to \mathcal{C}$, then it is sufficient for $\mathcal{C}$ to be a *symmetric* monoidal category, since then we have a natural transformation

$$(jX \multimap A) \otimes (jY \multimap B) \to (jX \otimes jY) \multimap (A \otimes B) \to j(X \otimes Y) \multimap (A \otimes B)\,.$$

Given a monoidal action, we get a natural transformation

$$(X \otimes Y).(A \otimes B) \to (X.A) \otimes Y.B \xrightarrow{\phi_{X,A} \otimes \phi_{Y,B}} A \otimes B\,,$$

which is multiplicative, inducing a functor $(\mathcal{C}\times\mathcal{C})/(\mathcal{M}\times\mathcal{M}) \to \mathcal{C}/\mathcal{M}$. Composing with the isomorphism from the start of this section, we get a functor $(\mathcal{C}/\mathcal{M}) \times (\mathcal{C}/\mathcal{M}) \to \mathcal{C}/\mathcal{M}$. It turns out that this gives $\mathcal{C}/\mathcal{M}$ the structure of a monoidal category and that in this case the functor $J\colon \mathcal{C} \to \mathcal{C}/\mathcal{M}$ is a strict monoidal functor.

This generalizes the case of a monoidal monad, whose the Kleisli category is always a monoidal category.

Now, if $\mathcal{C}$ is monoidal closed and the action of $\mathcal{M}$ on $\mathcal{C}$ is a *reader-type action*, then $\mathcal{C}/\mathcal{M}$ also inherits the monoidal closed structure from $\mathcal{C}$. Note that this may not be true for arbitrary actions, even for monads. For example, the Kleisli category for the powerset monad on **Set** (which is better known as the category of sets and relations) is monoidal closed, but with inner function space given by the Cartesian product, and not by the function space from **Set**.

3 Game Semantics for Probabilistic Algol

3.1 The discrete probability monad

Given a set X, we define a *discrete probability measure* on X to be a function $\mathbb{P}\colon \mathcal{P}(X) \to [0,1]$ satisfying the following conditions.

Empty set $\mathbb{P}(\varnothing) = 0$.

Countable additivity If $\{E_i\}_{i=0}^{\infty}$ is a sequence of pairwise-disjoint subsets of X, then

$$\mathbb{P}\left(\bigcup_{i=0}^{\infty} E_i\right) = \sum_{i=0}^{\infty} \mathbb{P}(E_i)\,.$$

Whole space $\mathbb{P}(X) = 1$.

If instead we have $\mathbb{P}(X) \leq 1$, we say that $\mathbb{P}$ is a *sub-probability measure*

Note that if A is countable, then μ may be entirelly specified by its action on singletons: if $A \subseteq X$ is a set, then we have

$$\mathbb{P}(A) = \sum_{a \in A} \mathbb{P}(\{a\})\,.$$

Given a set X, we define $P(X)$ to be the set of all discrete probability measures on X. Then we have natural maps

$$\delta_X\colon X \to P(X) \qquad\qquad m_X\colon P(P(X)) \to X$$

given by

$$\delta_X(x)(A) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases} \qquad\qquad \mathbb{P}_X(\tau)(A) = \sum_{\mathbb{P} \in P(X)} \tau(\{\mathbb{P}\})\mathbb{P}(A)$$

δ_X and m_X give P the structure of a monad on **Set**, which we will call the *discrete probability monad*. This is a discrete form of the monad developed in [15].

We want to exhibit this monad as the colimit of an action, as we did for the powerset monad earlier. We define a *discrete probability space* to be a pair $(X, \mathbb{P}_X)$, where X is a set and μ a discrete probability measure on X. We will normally refer to this pair using the set X. Given discrete probability spaces X and Y, we define a *measure-preserving map* from X to Y to be a function $f\colon X \to Y$ such that for all $A \subseteq Y$ we have $\mathbb{P}(X)(f^{-1}(A)) = \mathbb{P}_Y(A)$. It is clear that discrete probability spaces and measure-preserving maps form a category **DProb**.

Moreover, **DProb** is a monoidal category: given discrete probability spaces X and Y, we may define a probability measure on the Cartesian product $X \times Y$ by

$$\mathbb{P}_{X\times Y}(A) = \sum_{(x,y)\in A} \mathbb{P}_X(\{x\})\mathbb{P}_Y(\{y\})\,.$$

Then we have a natural isomorphism

$$P(X) \cong \varinjlim_{(Y,\mathbb{P}_Y)\colon \mathbf{DProb}^{op}} [Y, X]\,,$$

such that the monad structure on P is induced from the reader-style action of $\mathbf{DProb}^{op}$ on **Set** given by the monoidal functor $\mathbf{DProb} \to \mathbf{Set}$ that sends $(Y, \mathbb{P}_Y)$ to its underlying set Y.

Moreover, the category $\mathbf{DProb}^{op}$ is weakly filtered, and so, as in our earlier discussion, we feel confident using this action, rather than the monad P we have defined, to model discrete probability.

This action is very closely related to the action of $\mathbf{Surj}^{+,op}$ that we defined earlier; indeed, $\mathbf{Surj}^{+}$ is equivalent to the category of discrete probability spaces taking probabilities in the Boolean semiring $\{0, 1\}$ with countable summation given by maximum.

By cutting down the category **DProb** we can obtain slightly different notions of probability. For our purposes, we will be using the full subcategory **BProb** where the objects $(X, \mathbb{P}_X)$ are those that satisfy the following three additional conditions.

Countability The set X is countable.
Finite support There is a finite subset $E \subseteq X$ such that $\mathbb{P}_X(E) = 1$.

The action of $\mathbf{BProb}^{op}$ gives rise to a monad on the category of countable sets.

3.2 Probabilistic Algol

[14] gives a fully abstract game semantics for the language Probabilistic Algol (PA). The base language for PA is the language Idealized Algol (IA) from [7],

and it has been extended with a constants $\mathsf{coin}\colon \mathsf{bool}$. To make things a little more interesting, we will extend this further to incorporate an infinite family of constants $\mathsf{coin}_p\colon \mathsf{bool}$, where p ranges over the real interval $[0,1]$.

We give the language a small-step operational semantics. The rules for the deterministic terms may be derived from the operational semantics given in [7]; for example:

$$\frac{}{\langle s, v := n\rangle \longrightarrow \langle \langle s \mid v \mapsto n\rangle, \mathsf{skip}\rangle}$$

See also [16].

We add two small-step rules for the coin_p primitives:

$$\textsc{heads}_p\ \frac{}{\langle s, \mathsf{coin}\rangle_p \longrightarrow \langle s, \mathbb{t}\rangle} \qquad \textsc{tails}_{1-p}\ \frac{}{\langle s, \mathsf{coin}\rangle_p \longrightarrow \langle s, \mathbb{f}\rangle}\ .$$

These are the only nondeterministic parts of the operational semantics.

Given a small-step evaluation $\pi = M \to M_1 \to \cdots \to \mathtt{x}$ of a term M of ground type o, we define the *probability* $p(\pi)$ of π to be the product of all the indices q belonging to $\textsc{heads}_q$ or $\textsc{tails}_q$ rules.

This defines a discrete sub-probability measure on the set corresponding to the ground type o given by

$$\mathbb{P}(\{x\}) = \sum_{\substack{\text{evaluations}\\ \pi = M \to \ldots \mathtt{x}}} p(\pi)\,.$$

This is only a subprobability measure, because some evaluations may continue forever.

[14] shows how we can define other probabilistic terms using coin. Specifically, they define a term $\mathsf{polydie}\colon \mathbb{N} \to \mathbb{N}$, where the behaviour of $\mathsf{polydie}\, n$ is to return the result of tossing an n-sided die (with sides numbered 0 to $n-1$), with each outcome occurring with probability $1/n$. The definition of $\mathsf{polydie}\, n$ in terms of coin works as follows. We choose some number k such that $2^k \geq n$ and then toss the coin n times to give one of 2^k outcomes. If an outcome corresponds (under some fixed correspondence $n \hookrightarrow 2^k$) to a natural number below n, then we return that number; otherwise, we repeat the process until we get such a number.

This process will terminate with probability 1, and since each $m < n$ occurs with equal probability, they must all occur with probability $1/n$.

3.3 Extending a semantics for IA to a semantics for PA

Let $\mathcal{G}$ be a category that admits a fully abstract adequate semantics for Idealized Algol. See [7] for a list of the conditions that such a category should satisfy, and also for the best known example, the category of games and visible strategies.

As explained in [7], $\mathcal{G}$ should be Cartesian closed and there should be a lax monoidal functor $\mathbf{Set} \to \mathcal{G}$ that gives the denotation of the ground types $\mathtt{nat}$,bool and $\mathtt{com}$, together with their structural functions. We can therefore get

a lax monoidal functor $\mathbf{BProb} \to \mathcal{G}$ by composing this with the forgetful functor $\mathbf{BProb} \to \mathbf{Set}$.

This gives rise to a reader-style action of $\mathbf{BProb}^{op}/\mathcal{G}$, and the resulting category $\mathcal{G}/\mathbf{BProb}^{op}$ is therefore a symmetric monoidal closed category.

3.4 Testing morphisms and plays

The equivalence relation on morphisms is still too weak for our purposes. For example, if the boolean type $\mathbb{B}$ carries a Bernoulli distribution with coefficient p, then the diagonal map $\Delta\colon \mathbb{B} \to \mathbb{B} \times \mathbb{B}$ should be equivalent to the identity $\mathrm{id}\colon \mathbb{B} \times \mathbb{B} \to \mathbb{B} \times \mathbb{B}$ as terms of type $\mathbb{B} \times \mathbb{B}$, because the projections on to each component are equivalent. However, they are not yet equivalent in our category.

To improve the equivalence relation, we introduce the concept of *testing morphism* for an object A that is the denotation of an Idealized Algol term. This is a term of type $(\mathbb{N} \to A) \to A$ that is the denotation of an Idealized Algol term of the form

$$\mathtt{test}_{n_1,\cdots,n_k} = \lambda f.\mathtt{new}\ v \coloneqq 0\ \mathrm{in}\ f(v \coloneqq \mathtt{@}v + 1; \mathtt{case}\, v\ \_\ n_1\ \cdots\ n_k\, \Omega)\,,$$

where $n_1, \cdots, n_k$ is a sequence of numbers. The effect of this term is to call f, and to pass in the numbers $n_1, \cdots, n_k$ in turn, and diverging if f calls its argument more than k times. In game semantics, the sequence $n_1, \cdots, n_k$ is more easily recognized as a sequence of moves occurring in the game $\mathbb{N}$ on the left.

Given a morphism $\sigma\colon \mathbb{N} \to A$, we say that the sequence $n_1, \cdots, n_k$ is a *play* of σ if it is minimal (with respect to the prefix ordering) such that $\mathtt{test}_{n_1,\cdots,n_k}(\sigma)$ represents a non-diverging term. In that case, the *result* of the play is the strategy $\mathtt{test}_{n_1,\cdots,n_k}(\sigma)$ for A.

Now, if the set $\mathbb{N}$ carries a probability distribution, then this gives rise to a sub-probability distribution on the set of plays for σ, where the probability associated to the sequence $n_1, \cdots, n_k$ is the product of the probabilities of the n_i. This then induces a discrete probability distribution on the set of morphisms $1 \to A$.

We say that two morphisms $\sigma, \tau\colon \mathbb{N} \to A$ from 1 to A in $\mathcal{G}/\mathbf{BProb}$ are equivalent if they give rise to the same probability distribution on the set of strategies for A. This equivalence relation refines our earlier one.

3.5 Single-threaded morphisms

Our category is symmetric monoidal closed, but not yet Cartesian closed. A trick that is often used in Game Semantics (see, for example, [8] and [14]) is to pass to a category of *single-threaded* strategies; this is equivalent to cutting down to those morphisms that are comonoid homomorphisms.

Note that, in our category, every object has a natural commutative comonoid structure that arises from diagonal $\Delta\colon A \to A \times A$ in the original category. We

say that a morphism $\sigma\colon A \to B$ is *single-threaded* if it is a homomorphism of the comonoid structures on A and B; i.e., if the following square commutes.

$$\begin{array}{ccc} A & \xrightarrow{\sigma} & B \\ {\scriptstyle\Delta}\downarrow & & \downarrow{\scriptstyle\Delta} \\ A \times A & \xrightarrow{\sigma\times\sigma} & B \times B \end{array}$$

Since the category of commutative comonoids in a symmetric monoidal category is automatically Cartesian closed (see, for example, [17]), this gives us a Cartesian closed category.

The deterministic terms are all comonoid homomorphisms. As long as we have applied the equivalence relation outlined above, the natural morphisms $\phi_{X,A}$ are also comonoid homomorphisms.

3.6 Denotational semantics and adequacy

We may now inductively build up a denotational semantics for PA. The deterministic terms may all be defined inside the base category $\mathcal{G}$ and transported into our new category via the natural inclusion. The probabilistic term coin_p is given by the natural morphism $\phi_{B_p,1}$, where B_p is a Bernoulli space with coefficient p. That is to say, it is given by the pair

$$((\{H,T\}, \mu(H) = p, \mu(T) = 1-p), 1 \to (\{H,T\} \to \mathbb{B}))$$

where the function on the right sends H to $\mathbb{t}$ and T to $\mathbb{f}$.

Note now that the denotation of a term $M = C[\mathsf{coin}_{p_1}, \cdots, \mathsf{coin}_{p_j}]T$ is given by the following morphism.

$$\mathbb{B}^j \xrightarrow{[\![b_1,\cdots,b_j \vdash C[b_1,\cdots,b_j]]\!]} [\![T]\!] \ ,$$

where the distribution on the product $\mathbb{B}^j$ is the product of Bernoulli distributions.

Fix an injection $r\colon \mathbb{B}^j \to \mathbb{N}$ (for instance, the one given by ibinary representation) and suitable projection functions $\mathrm{pr}_i\colon \mathbb{N} \to \mathbb{B}$ such that $\mathrm{pr}_i(r(v))$ is the i-th entry of v. Let $\mathsf{multicoin}\colon \mathtt{nat}$ be the term (depending on the values of the p_i) given by

$$(r\ \mathsf{coin}_{p_1}\ \cdots\ \mathsf{coin}_{p_j})\,.$$

Now the operational semantics of $\mathrm{pr}_i(\mathsf{multicoin})$ is the same as the operational semantics of coin_{p_i}, since the projections from the product are probability-preserving. This means that any evaluation π of the term M may be pegged to an evaluation of the term formed by replacing each instance of coin_{p_i} in M with $\mathrm{pr}_i(\mathsf{multicoin})$.

Proposition 3 (Computational Adequacy). *Let $M = C[\mathit{multicoin}]\colon \mathtt{com}$ be a term of Idealized Algol, where C is a multi-holed deterministic IA context. Let*

p be the sum of the probabilities of all evaluation sequences of M that terminate at skip.

The denotation of M is a morphism $\mathbb{N} \to \mathbb{C}$ *as outlined above.*

Then we get an induced sub-probability distribution on the set of morphisms $1 \to A$. *The adequacy result is that p is the same as the induced probability of the morphism* $[\![\mathsf{skip}]\!]$.

Proof. Each terminating evaluation sequence of M corresponds to a finite sequence $n_1, \cdots, n_k$ of the different values that multicoin takes on over the course of the evaluation of the term. By inspection, we may peg this evaluation of M exactly to the evaluation of the term $\mathtt{test}_{n_1,\cdots,n_k}(\lambda n.C[n])$ in pure deterministic Idealized Algol, and therefore this IA term must converge. By the Adequacy result for Idealized Algol [7], an IA term converges to skip if and only if its denotation is $[\![\mathsf{skip}]\!]$. Therefore, there is bijective matching between terminating evaluation sequences of M and plays of the morphism $[\![M]\!]$, with each one getting the same probability. The result follows.

3.7 Full Abstraction

In order to prove full abstraction for our model, we will define a further equivalence on morphisms. Say that two morphisms $f, g \colon A \to B$ are *intrinsically equivalent* if and only if for all morphisms $\alpha \colon (A \to B) \to \mathbb{C}$, $\alpha(f)$ and $\alpha(g)$ are probabilistically equivalent in the sense given above. Then our full abstraction result becomes:

Proposition 4 (Full Abstraction). *Let* $M, N \colon T$ *be two PA terms. Then M and N are observationally equivalent if and only if their denotations* $[\![M]\!]$ *and* $[\![N]\!]$ *are intrinisically equivalent.*

One direction (soundness) follows immediately from our Computational Adequacy result. To prove the other direction, it suffices to show that any compact strategy is definable [18], where we say that a morphism $\sigma \colon A \to (X \to B)$ from A to B is *compact* if it is compact when considered as a morphism in $\mathcal{G}$. We prove this using a factorization result:

Lemma 1 (Deterministic factorization). *Let* $\sigma \colon A \to B$ *be a morphism in* $\mathcal{G}/\mathbf{BProb}^{op}$. *Then* σ *may be factorized as* $Det(\sigma)(\mathsf{multicoin})$, *where* $Det(\sigma) \colon \mathbb{N} \to A \to B$ *is a strategy in the image of the inclusion functor* $J \colon \mathcal{G} \to \mathcal{G}/\mathbf{BProb}^{op}$, *and* multicoin *is defined for some values* $p_1, \cdots, p_k$.

Proof. Treat σ as a morphism $A \to X \to B$ in $\mathcal{G}$. Since X is finitely supported, there is a probability-preserving function $\mathbb{N} \to X$, for some finitely supported distribution on $\mathbb{N}$. So σ is equivalent to a morphism $A \to \mathbb{N} \to B$, and then the factorization is automatic.

References

1. E. Moggi, in *[1989] Proceedings. Fourth Annual Symposium on Logic in Computer Science* (1989), pp. 14–23. DOI 10.1109/LICS.1989.39155
2. M. Spivey, Science of Computer Programming **14**(1), 25 (1990). DOI https://doi.org/10.1016/0167-6423(90)90056-J. URL http://www.sciencedirect.com/science/article/pii/016764239090056J
3. P. Wadler, in *Proceedings of the 19th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (ACM, New York, NY, USA, 1992), POPL '92, pp. 1–14. DOI 10.1145/143165.143169. URL http://doi.acm.org/10.1145/143165.143169
4. P. Wadler, in *Proceedings of the 1990 ACM Conference on LISP and Functional Programming* (ACM, New York, NY, USA, 1990), LFP '90, pp. 61–78. DOI 10.1145/91556.91592. URL http://doi.acm.org/10.1145/91556.91592
5. M. Hyland, J. Power, Electronic Notes in Theoretical Computer Science **172**, 437 (2007). DOI https://doi.org/10.1016/j.entcs.2007.02.019. URL http://www.sciencedirect.com/science/article/pii/S1571066107000874. Computation, Meaning, and Logic: Articles dedicated to Gordon Plotkin
6. F.E.J. Linton, in *Proceedings of the Conference on Categorical Algebra*, ed. by S. Eilenberg, D.K. Harrison, S. MacLane, H. Röhrl (Springer Berlin Heidelberg, Berlin, Heidelberg, 1966), pp. 84–94
7. S. Abramsky, G. McCusker, Theor. Comput. Sci. **227**(1-2), 3 (1999). DOI 10.1016/S0304-3975(99)00047-X. URL http://dx.doi.org/10.1016/S0304-3975(99)00047-X
8. R. Harmer, G. McCusker, in *Proceedings. 14th Symposium on Logic in Computer Science (Cat. No. PR00158)* (1999), pp. 422–430. DOI 10.1109/LICS.1999.782637
9. D.R. Ghica, in *Proceedings of the 32Nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (ACM, New York, NY, USA, 2005), POPL '05, pp. 85–97. DOI 10.1145/1040305.1040313. URL http://doi.acm.org/10.1145/1040305.1040313
10. P.A. Melliès. Parametric monads and enriched adjunctions (2012). Preprint on webpage at https://www.irif.fr/~mellies/tensorial-logic/8-parametric-monads-and-enriched-adjunctions.pdf
11. A.L. Smirnov, Journal of Mathematical Sciences **151**(3), 3032 (2008). DOI 10.1007/s10958-008-9013-7. URL https://doi.org/10.1007/s10958-008-9013-7
12. S.y. Katsumata, Graded monads and semantics of effect systems (2016). URL http://csl16.lif.univ-mrs.fr/planning/qslc/talks/katsumata/. QSLC 2016
13. C. Lüth, N. Ghani, in *Category Theory and Computer Science*, ed. by E. Moggi, G. Rosolini (Springer Berlin Heidelberg, Berlin, Heidelberg, 1997), pp. 69–86
14. V. Danos, R. Harmer, in *Proceedings Fifteenth Annual IEEE Symposium on Logic in Computer Science (Cat. No.99CB36332)* (2000), pp. 204–213. DOI 10.1109/LICS.2000.855770
15. M. Giry, in *Categorical Aspects of Topology and Analysis*, ed. by B. Banaschewski (Springer Berlin Heidelberg, Berlin, Heidelberg, 1982), pp. 68–85
16. J. Laird, G. Manzonetto, G. McCusker, M. Pagani, in *Logic in Computer Science, Symposium on(LICS)*, vol. 00 (2013), vol. 00, pp. 301–310. DOI 10.1109/LICS.2013.36. URL doi.ieeecomputersociety.org/10.1109/LICS.2013.36
17. P. Freyd, P. O'Hearn, A. Power, M. Takeyama, R. Street, R. Tennent, Theoretical Computer Science **228**(1), 49 (1999). DOI https://doi.org/10.1016/S0304-3975(98)00354-5. URL http://www.sciencedirect.com/science/article/pii/S0304397598003545

18. P.L. Curien, Electron. Notes Theor. Comput. Sci. **172**, 301 (2007). DOI 10.1016/j.entcs.2007.02.011. URL http://dx.doi.org/10.1016/j.entcs.2007.02.011