Electronic Notes in Theoretical Computer Science **14** (1998)
URL: http://www.elsevier.nl/locate/entcs/volume14.html 12 pages

# Monads in Semantics

Philip S. Mulry [1]

*Department of Computer Science*
*Colgate University*
*Hamilton, NY 13346*

**Abstract**

In this extended abstract we provide a very brief overview of the notion of a monad along with some examples of applications to programming language semantics. The treatment is by no means exhaustive but rather chooses examples and results that either illustrate the wide variety of uses of this abstract tool or which bear some connection to other work presented at the workshop. The abstract begins with some preliminary definitions and examples, proceeds to categories of algebras and ends with some results and examples of the author using monadic lifting.

## 1 Introduction

This extended abstract provides a survey of some aspects of the use of monads in the semantics of programming languages. It became apparent during the course of this workshop that while monads are used in many different guises in the formal foundations of software systems, they often appear explicitly in only limited formats. In particular the potential exploitation of monads in categories of algebras appears to remain relatively untapped. Also apparent at the workshop was the apparent lack of familiarity, among investigators, of each other's work using monadic tools. One goal of this abstract then is to begin to bridge this gap. While the brevity of the abstract requires that the exposition be incomplete, touching only briefly on a few aspects of the topic, it is hoped nonetheless that this work may spur additional communication and interaction between investigators using these tools.

It is an incorrect, but somewhat widely held notion, that the use of monads in programming language semantics is a relatively recent phenomena of the past decade or so, often associated with work that began emerging in the late 1980's on the semantics of functional languages. While this work has been both important and influential, in fact monads and algebras have had a long

[1] This research was partially supported by NSF Grant CCR-9506383

and rich history of application in semantics that can be traced back to at least the early work of Lawvere and Manes on algebraic theories, and including the work of any number of investigators including Arbib, Benson, Burstall, Ehrig, Eilenberg, Goguen, Lambek, Thatcher, Wagner, Wand, and Wright. Work through the past two decades on categorical models of computation made explicit use of monads and algebras including the work of Freyd, Hyland, Plotkin, Moggi, Mulry, Rosolini, Scott, and Smyth, to name just a few. We don't intend to pursue a historical account here but rather will simply highlight some of the developments in this area, both recent and otherwise.

Of necessity this author has chosen just a few examples illustrating the ways monads are used in semantics. There is no attempt to be complete; instead an emphasis is placed on applications and examples that may be either less well known or that relate in some fashion to themes raised in this workshop. Also, no attempt to cite the related work of every workshop participant was made. The abstract begins with some preliminary definitions and examples, proceeds to categories of algebras, and ends with some results and examples of the author using monadic lifting. A more detailed exposition of these and related results, including proofs, can be found in [18] and [19].

## 2 Preliminaries

Most expositions of monads begin with a definition. We choose instead to start with a motivating canonical example.

**Example 2.1** Let $\mathsf{C}$ be the category of sets and fix a (small) monoid $(M, e)$. Let $H$ be the endofunctor on $\mathsf{C}$ defined by $H(A) = M \times A$. It is immediate that there are two natural transformations $\eta, \mu$ defined as follows: $\eta_A : A \to M \times A$ where $\eta_A(a) = (e, a)$ and $\mu_A : M \times (M \times A) \to M \times A$ where $\mu_A(m, (n, a)) = (mn, a)$. It is immediate that the following equations hold:

1) $(em, a) = (m, a)$
2) $(me, a) = (m, a)$
3) $(m(np), a) = ((mn)p, a)$

Abstracting from this example we get the usual definition of a monad.

**Definition 2.2** A monad $(H, \mu, \eta)$ on category $\mathsf{C}$ consists of an endofunctor $H$ and two natural transformations $\eta : id \to H$ and $\mu : H^2 \to H$ satisfying

1) $\mu \circ \eta_H = id_H = \mu \circ H\eta$
2) $\mu \circ \mu_H = \mu \circ H\mu$

**Example 2.3** Monads arise in many ways including those arising in the presence of adjoint pairs. Specifically any adjoint pair $F, G$ gives rise to the monad $(GF, \eta, G\epsilon F)$. It is well known that monads in turn give rise to adjunctions. The initial and final such adjunctions arise in the context of Kleisli and Eilenberg -Moore categories which we will consider in more detail shortly.

Next we look at two very familiar examples of adjunctions, a free construction and a reflection, leading to the formation of monads.

**Example 2.4** Let $F$ be the free monad functor from SET to MON the category on monoids with forgetful right adjoint $U$. The monad formed by this adjunction is Kleene star that acting on set $X$ produces $X^*$ the set of strings on alphabet $X$. $\eta(x) =' x'$ coerces a character into a string of length one and $\mu$ acting on a string of words concatenates the words into a single string.

**Example 2.5** Let $R$ be the reflection from the category CUSL of complete upper semilattices to Dom the category of domains. The reflection $R$ is left adjoint to the inclusion and the corresponding monad $H$ acts as follows: for complete upper semilattice $P$, $H(P) =$ ideals on $P$, $\eta(a) =\downarrow (a)$ and $\mu$ is just union.

**Example 2.6** Monads arise in many other settings as well. Sheaf models, for instance, appear in programming language semantics both past and present. Sheaves over a category C are examples of a topos. Every topos comes equipped with an internal truth value object $\Omega$. This object is utilized to generate not only the internal logic of the sheaf model but also to interpret partial maps by automatically generating a partial map classifier $H$. H is not only a monad, it is also strong and computational. In [14] it is shown that there are many other such partial map classifiers existing in a topos setting. For example in REC, the Recursive Topos, there are an infinite number of such monads that reflect the computational complexity of the arithmetic hierarchy. Further, the notion of partial map classifier is not restricted solely to toposes. A further exposition of these matters can be found in [17].

**Example 2.7** If $P$ is the powerset functor on SET, then $P$ is a monad where for any set $A$, $\eta_A(a) = \{a\}$ and $\mu_A\{A_i\} = \cup A_i$

**Example 2.8** Suppose we add a monoid action, $*$, to set $A$ from Example 2.1. We have then $* : M \times A \to A$ where

1) $e * a = a$
2) $m * (n * a) = (mn) * a$

We have arrived at the notion on an Eilenberg-Moore algebra.

**Definition 2.9** Let $H$ be a monad on C. An object $A$ in C is an Eilenberg-Moore(*E-M*) algebra if there exists a structure map $h : HA \to A$ in C so that

1) $h \circ \eta_A = id_A$
2) $h \circ Hh = h \circ \mu_A$

There is a potential source of confusion in the use of the algebra terminology. It has also become common in the presence of a monad $H$ to let any pair $(A, q)$ where $q : HA \to A$ is an arrow in C denote an algebra, even if

equations 1) and 2) above do not hold, and further, even when $H$ is simply an endofunctor of $\mathsf{C}$. To avoid such confusion we shall always utilize the prefix *E-M* to denote Eilenberg-Moore algebras.

**Example 2.10** Continuing with Example 2.4 where the monad is Kleene star, $(\ )^*$, *E-M* algebras correspond to monoids. Multiplication on algebra object $(X, h)$ is determined by the structure map $h$, namely the monoid product of x and y is just $h('xy')$ while the identity element is simply the image under $h$ of the empty string.

**Example 2.11** Continuing with Example 2.7 where $P$ is the powerset monad, *E-M* algebras correspond to complete semilattices. Order on algebra object $(A, h)$ is determined as follows: for $A_0 \subseteq A, h(A_0) = \bigvee A_0$. In particular for a,b in $A$, $a \leq b$ iff $h\{a, b\} = b$.

## 3 Categories of Algebras

A key observation in the use of monads is the existence of two special categories of algebras: the Kleisli and Eilenberg-Moore categories of algebras. They represent initial and final solutions to the problem of finding adjunctions that generate a given monad. In this section we briefly describe just a few of the recent utilizations of these categories in semantics. We begin with Kleisli categories.

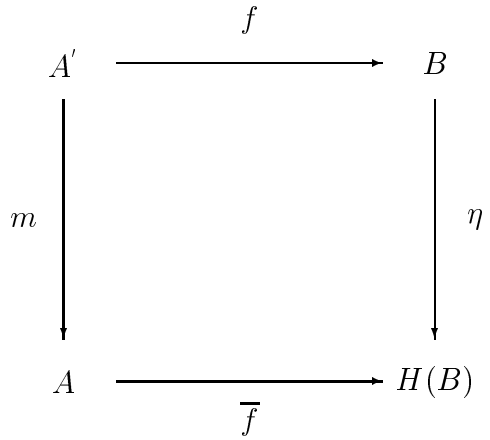
**Definition 3.1** Let $(H, \eta, \mu)$ be a monad on category $\mathsf{C}$. The Kleisli category $\mathsf{C}_H$ has the same objects as in $\mathsf{C}$. Arrows from $A$ to $B$ in $\mathsf{C}_H$ correspond to arrows $A \to HB$ in $\mathsf{C}$. The definition of arrows in Kleisli makes composition of arrows non-trivial. If f is an arrow from $A$ to $B$, and g an arrow from B to C, both in $\mathsf{C}_H$, the composition corresponds to the arrow $\mu_B \circ Hg \circ f$ in $\mathsf{C}$. It is easy to check that this composition is well defined.

There is a standard inclusion functor $i_H : \mathsf{C} \to C_H$. It is well known that $i_H$ has a right adjoint, $i_H \dashv R_H$, that the monad formed by the adjunction is just $H$ and that it is the initial such adjunction generating $H$. What appears to be less well known is that the Kleisli category on $H$ is equivalent to the category of free Eilenberg-Moore algebras on $H$. We return to this after we introduce *E-M* algebras in detail. In the interim we describe several examples that utilize Kleisli categories. Again this list is far from inclusive.

**Example 3.2** Monads can be used to model partial maps. A domain structure on $\mathsf{C}$ is a family of subobjects, $M(A)$, for each object $A$, satisfying certain conditions. Given a domain structure it is easy to construct $(\mathsf{pC}, M)$, the category of partial maps for $M$. Objects coincide with those from $\mathsf{C}$ and an arrow from $A$ to $B$ in $\mathsf{pC}$ consists of a pair of maps (m,f) in $\mathsf{C}$ where $m : A' \to A$ is in $M(A)$ and f is a map $f : A' \to B$ in $\mathsf{C}$.

$H$ is a partial map classifier (*pmc*) for category $\mathsf{C}$ with domain structure $M$ if $H$ is an endofunctor of $\mathsf{C}$, so that for any object $B$, a mono map $\eta : B \stackrel{\eta_B}{\to} HB$

in $M(HB)$ exists, satisfying the following universal property: for any partial map $(A, m) \xrightarrow{f} B$ there exists a unique total map $A \xrightarrow{\overline{f}} HB$ making the diagram a pullback:




The following result is of immediate interest. If $H$ is a *pmc* on cartesian category $\mathsf{C}$, then $H$ is a strong and commutative monad. Further if $H$ is a *pmc* on $\mathsf{C}$, then $\mathsf{pC}$ is equivalent to $\mathsf{C}_H$, the Kleisli category for $H$. Quite a bit more can be said on this subject. The interested reader can find further details in [17].

**Example 3.3** The $\lambda_p$ calculus of Moggi is sound and complete with respect to interpretation in partial cartesian categories, *pcccs*. Each *pccc* $\mathsf{pC}$ has a generic partial map classifier monad $H$ so that $\mathsf{pC}$ is equivalent to $\mathsf{C}_H$. Thus terms are interpreted as maps in $\mathsf{C}_H$.

**Example 3.4** In [1] the authors utilize the dual notion to a monad, namely a comonad, to build an intensional semantics. Starting with category $\mathsf{C}$ equipped with a computational comonad $H$, arrows in the Kleisli category of the comonad are represented by arrows $HA \to B$ in $\mathsf{C}$, i.e. arrows from input computations over data type $A$ to values of data type $B$. These are referred to as algorithms to emphasize their computational content. In this way one can distinguish between two different algorithms that are extensionally equivalent.

**Example 3.5** The notion of a monad has been used extensively in work on the semantics of programming languages, most notably in the work of Moggi [12] and Wadler [25]. Given a monad $H$ on category $\mathsf{C}$, $HA$ is interpreted to be an object of computations on $A$. A program can then be viewed as a map from values to computations, which can be represented as an arrow $A \to B$ in $\mathsf{C}_H$. Simple example of monads over the category $\mathsf{SET}$ include

non-determinism: $HA = P(A)$ where $P$ is the powerset monad

side-effects: $HA = (A \times S)^S$ where S is the set of stores

continuations: $HA = R^{R^A}$ where R is a set of results

partiality: $HA = A_\perp$ where $A_\perp$ stands for $A$ with a new bottom.

exceptions: $HA = A + E$ where E is a set of exceptions.

More recently monad constructors or transformers, first proposed by Moggi, have been introduced to design fully modular interpreters [9].

**Example 3.6** As mentioned earlier, sheaf models, and in fact all toposes, are rich in monads in part because every topology generates a corresponding partial map classifier monad. These are not the only ones. Other partial map classifiers may be generated by special subobjects of the truth value object $\Omega$ which may not be associated with any fixed topology. In the Recursive Topos $\mathsf{REC}$ for example, there are an infinite number of pmc monads corresponding to different levels in the arithmetic hierarchy. A very special example of such a monad denoted $\widetilde{(\ )}_{re}$ was observed in $\mathsf{REC}$ [14], with the equivalent monad, denoted $\Sigma$, defined in the Effective Topos $\mathsf{EFF}$ [5]. This monad was useful not only in modeling partial maps but also in interpreting terms in the topos. Thus for example the construction of the effective real numbers in either topos utilizes $\widetilde{(\ )}_{re}$ or $\Sigma$. This monad also appears in work on $\mathsf{PER}$, the category of partial equivalence relations and especially in the work on $\mathsf{ExPERs}$ [4] where it plays a critical role in the discussion. For example the initial algebra for $\Sigma$ is the effective vertical natural numbers object. It is a fixed point object, in fact it is an invariant object(i.e. an initial algebra isomorphic to its final coalgebra), and so also plays an important role in interpreting fixed points.

**Example 3.7** Comonads also play an important role in linear logic. For example in [22] an equivalence between linear logics with ! and Girard categories is established where a Girard category is a linear category with a comonad !. More recent work in [6] on the semantics of weakening and contraction further illustrates the connections between linear logic and comonads.

**Definition 3.8** Let $(H, \eta, \mu)$ be a monad on category $\mathsf{C}$. The *E-M* category of $H$-algebras, denoted $\mathsf{C}^H$, has

- Objects: *E-M* algebras $(A, a)$
- Arrows: an arrow from $(A, a)$ to $(B, b)$ in $\mathsf{C}^H$ is an arrow $f : A \to B$ in $\mathsf{C}$ so that $b \circ Hf = f \circ a$.

There is a standard forgetful functor $U_H : \mathsf{C}^H \to \mathsf{C}$. $U_H$ has a left adjoint, $F_H \dashv U_H$, which on object $A$ in $\mathsf{C}$ creates the free algebra $(HA, \mu_A)$. The monad formed by the adjunction is again just $H$ and it is the final such adjunction generating $H$. In particular, there always exists a comparison functor $G : \mathsf{C}_H \to \mathsf{C}^H$ which is a map of the corresponding adjunctions.

**Example 3.9** Returning to the powerset monad $P$ on $\mathsf{SET}$, *E-M* algebras $(X, x)$ and $(Y, y)$ are complete semilattices. Algebra maps f from $(X, x)$ to $(Y, y)$ are set functions so that $y \circ Pf = f \circ x$. This in turn forces f to preserve $\bigvee$ and therefore order as well. $\mathsf{SET}^P$ then is the category of complete semilattices and sup-preserving maps.

**Example 3.10** Let BDOM be the category of (possibly) bottomless domains and continuous maps. If $H$ is the "lift" monad that adds a new bottom, $\uparrow$, then $H$-algebras correspond to domains with bottom where for any *E-M* algebra (D,d), $d(\uparrow) = \perp_D$. This is easy to see since $\uparrow \leq \eta(d)$ for all d in BDOM and thus $d(\uparrow) \leq d\eta(d) = d$. In particular, algebra maps enforce strictness and thus $\mathsf{BDOM}^H \cong \mathsf{SDOM}$ the category of domains with strict maps.

**Example 3.11** For a given monad $(H, \eta, \mu)$, the Kleisli category is equivalent to the category of free Eilenberg-Moore algebras on $H$. This equivalence can be seen by using the comparison functor G. For $f : A \to HB$ an arrow in $\mathsf{C}_H$, $G(f) : HA \to HB$ is defined to be $G(f) = \mu_A \circ H(f)$. It is easy to show that $G(f)$ is an algebra map and that every map of free algebras arises in this way. A particular instance of this equivalence is the well known equivalence between the category of (possibly) bottomless domains and partial maps and the category of domains and strict maps described in the previous example.

**Example 3.12** In [15] the connection between strong monads, *E-M* algebras and fixed points is made. For any *ccc* C with strong monad and fixed point object Z, there exists a fixed point combinator $FIX$ which acts on *E-M* algebras. The combinator is an algebraically strong dinatural transformation $(\ )^{(\ )} \stackrel{\cdot\cdot}{\to} (\ )$ with factorization utilizing the fixed point object. For example if C is the category of complete lattices with maps preserving nonempty sups and infs then monad $H = (\ )_\perp$ generates the vertical fixed point object and the corresponding algebraically strong dinatural transformation is just *fix*, the least fixed point operator. If $H = (\ )_\top$ is the strong monad that adds a new top to the lattice, then there is a new fixed point object and the corresponding algebraically strong dinatural transformation is just *FIX*, the greatest fixed point operator.

**Example 3.13** Recently Val Tannen has utilized monads to describe typed systems in data base query languages [24]. Here the idea is to model aggregate types through enriched algebras. For example, if $Coll$ is the collection functor on SET, where $Coll$ may mean $Set, Bag, List$ etc. then with $\eta_A = sng_A$ generating singletons and $\mu_A = flatten_A$, $Coll$ is a monad. Taking a standard numerical data type such as Int, then $(Int, aggmax)$ is a $Coll$ algebra for $Coll = Set, Bag, List$. $(Int, aggsum)$ is a $Coll$ algebra for $Coll = Bag, List$ but not for $Set$. Here the equation $aggsum \circ Coll(aggsum) = aggsum \circ flatten$ fails for a set of the form $\{\{1,2\},\{2,3\}\}$.

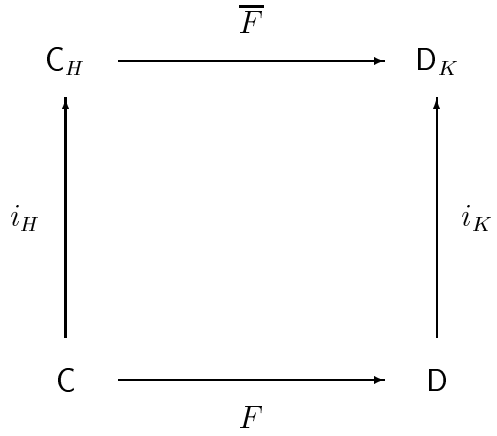
## 4 Lifting Tools

As the previous section indicates it is fruitful to exploit the presence of monads in the corresponding categories of algebras. In particular it is useful indeed to know when a computational process, such as a type constructor, in the form of a functor lifts to the corresponding categories of algebras. Also of importance is understanding when adjunctions might lift. In this section we present some

results utilizing liftings of functors. A more detailed description can be found in [18].

Consider monads $(H, \eta, \mu)$ and $(K, \rho, \nu)$ on categories $\mathsf{C}$ and $\mathsf{D}$ respectively. Let $F$ be a functor $F : \mathsf{C} \to \mathsf{D}$. The notion of the lifting of a functor $F$ exists for both Eilenberg-Moore and Kleisli categories. Let $i_H$, $i_K$ denote the inclusion functors into Kleisli categories.

**Definition 4.1** A functor $\overline{F} : \mathsf{C}_H \to \mathsf{D}_K$ is a Kleisli lifting of $F$ if $\overline{F} \circ i_H = i_K \circ F$ or equivalently that the following diagram commutes.




One can specify conditions that ensure such liftings exist.

**Theorem 4.2** *For $\mathsf{C}$, $\mathsf{D}$, $H, K, F$ as above, functors $\overline{F} : \mathsf{C}_H \to \mathsf{D}_K$ which are liftings are in 1-1 correspondence with natural transformations of the form $\lambda : FH \to KF$ that satisfy the following*

1) $\lambda \circ F\eta = \rho_F$
2) $\nu_F \circ K\lambda \circ \lambda_H = \lambda \circ F\mu$.

**Proof.** See [10] and [16] where further references can be found. □

In a similar fashion one can define Eilenberg-Moore($E$-$M$) liftings $F^* : \mathsf{C}^H \to \mathsf{D}^K$ to Eilenberg-Moore($E$-$M$) categories of algebras where the equation $F \circ U_H = U_K \circ F^*$ holds for forgetful functors $U_H, U_K$. Such liftings are ensured by the existence of natural transformations of the form $\sigma : KF \to FH$ satisfying equations similar to those for $\lambda$ above. See [7].

**Notation:** We will denote Kleisli lifts of $F$ by $\overline{F}$ and Eilenberg-Moore lifts by $F^*$. Again we are supposing that functor $F : \mathsf{C} \to \mathsf{D}$ exists with accompanying monads $H$ and $K$.

Examples of Kleisli liftings arise in many different settings. Below we present just a few such examples.

**Definition 4.3** A monad on a symmetric monoidal category $\mathsf{C}$ is strong if there exists a natural transformation $\lambda_{A,B} : HA \otimes B \to H(A \otimes B)$ satisfying 1) and 2) below. If, in addition, $H$ is monoidal then we say $H$ is commutative.

1) $\lambda_{A,B} \circ \eta_A \otimes B = \eta_{A \otimes B}$.

2) $\mu_{A \otimes B} \circ H\lambda_{A,B} \circ \lambda_{HA,B} = \lambda_{A,B} \circ \mu_A \otimes B$

**Example 4.4** The notion of monadic strength arises naturally in a lifting context. Let $\mathsf{C}$ be a cartesian category with monad $H$. The product functor $\_ \times B$ lifts to the Kleisli category $\mathsf{C}_H$ exactly when $H$ is strong. Likewise if $H$ is commutative then the bifunctor $\_ \times \_$ lifts.

**Example 4.5** Suppose $\mathsf{C}$ and $\mathsf{D}$ agree, $\mathsf{C}$ has the trivial identity monad and $K$ becomes $H$. Then for a given endofunctor $T$ of $\mathsf{C}$, the natural transformation $\eta_T : T \to HT$ satisfies the equations of Theorem 4.2 and so a lifting $\overline{T} : \mathsf{C} \to \mathsf{C}_H$ exists. Conversely if $\overline{T}$ exists then $\overline{T} = i_H \circ T$ and so $\lambda$ must be $\eta_T$. In particular when $T$ is the identity, $\lambda$ is just the unit of the monad $H$, $\eta$, and $\overline{T} = i_H$.

**Example 4.6** Suppose $\mathsf{C}$ and $\mathsf{D}$ agree, $K$ is the trivial identity monad and $T$ is the monad $H$ itself on $\mathsf{C}$. In this case $H$ has a lifting and the corresponding natural transformation is just $\mu : \lambda : H^2 \to H$. The equations of Theorem 2.2 hold and reduce to the identities $\mu \circ H\eta = id_H$ and $\mu \circ \mu_H = \mu \circ H\mu$. The lifting $\overline{H}$ is the right adjoint to $i_H$, namely $G_H$.

**Example 4.7** Lifting distributes over composition, *i.e.* $\overline{S} \circ \overline{T} = \overline{ST}$. Utilizing the two previous examples we can generate the comonad associated to $H$ via a lifting, namely as the lift $\overline{H} = \overline{id \circ H} = i_H \circ G_H$. This observation also motivates our later use of the notation $\overline{H}$ to denote this comonad.

**Example 4.8** Returning to Example 3.4 and using duality, intensional semantics can be presented via liftings. Beginning with a comonad $H$ the two basic functors *alg* and *fun* that appear in [1] arise as liftings. Consequently their composition is also a lifting which represents identity only up to extensional equivalence. Likewise the condition defining a computational comonad is simply the existence of a natural trasformantion $\sigma$ guaranteeing the existence of a lifting. Resulting equations and properties such as the existence of adjoints then arise from the general theory [16].

**Example 4.9** Kleisli liftings also play a role in partial map semantics. The following result can be derived from the general theory of liftings (see [16]). Let $\mathsf{C}$ be a model of the computational lambda calculus, $\lambda_c$, in the sense of [11] where monad $H$ is cartesian. If $\mathsf{C}$ has equalizers of coreflexive pairs then $\mathsf{C}$ is a ccc.

We now turn our attention to Eilenberg-Moore liftings.

**Example 4.10** If $\mathsf{C}$ has products then $\mathsf{C}^H$ inherits them. This can be easily proven directly or by observing that the *E-M* lifting of the bifunctor exists where $\sigma = (\mathrm{fst}, \mathrm{snd})$.

We are particularly interested when not just functors but adjoint pairs lift to *E-M* categories. It is well known, for example, that even when a category

$\mathsf{C}$ is cartesian closed, the corresponding category of $E-M$ algebras $\mathsf{C}^H$ need not be. A reasonable question to ask is whether we can come close, i.e. is there a corresponding category, whose objects consist of the $E-M$ algebras, that is cartesian closed. The answer, under mild conditions, is yes. Further this result is just a special case of a far more general process that permits the lifting of adjoint pairs. We state the result next. A more detailed description can be found in [18]. As indicated earlier, if $H$ is a monad then $\overline{H}$ refers to its corresponding comonad.

**Theorem 4.11 (Basic Result)** *Consider monads $(H,\eta,\mu)$ and $(K,\rho,\nu)$ on categories $\mathsf{C}$ and $\mathsf{D}$ respectively and let $F:\mathsf{C}\to\mathsf{D}$ be a functor.*

1) *If $F$ has a right adjoint $G$, then the Kleisli lifting $\overline{F}$ exists if and only if the E-M lifting $G^*:\mathsf{D}^K\to\mathsf{C}^H$ exists.*
2) *If the Kleisli lifting $\overline{F}$ of $F$ exists and $\mathsf{D}^K$ has coequalizers of reflexive pairs, then there always exists a functor (but not a lifting!) $\widetilde{F}:\mathsf{C}^H\to\mathsf{D}^K$.*
3) *In this case, there exists an isomorphism $\tau:\widetilde{F}F^H\to F^KF$ where $F^H,F^K$ are the free algebra functors.*
4) *Further, if $F$ has a right adjoint $G$, then $\widetilde{F}\dashv G^*$.*
5) *If the E-M lifting $F^*$ also exists then then there exists an adjoint pair between Kleisli categories of comonads, $\overline{F^*}\dashv\overline{G^*}$, where $\overline{F^*}:(\mathsf{C}^H)_{\overline{H}}\to(\mathsf{D}^K)_{\overline{K}}$. Thus we have achieved a lifting of adjoint pairs.* □

**Example 4.12** If $\mathsf{C}$ is the category $\mathsf{BDOM}$ of (possibly) bottomless Scott domains with $H$ the "lift" monad, $(\ )_\perp$ then since $H$ is a *pmc* it is strong and commutative. $\mathsf{C}^H$ is Scott domains and strict maps, $\mathsf{SDOM}$. If $F=\_\times\_$ then $\widetilde{F}$ is the usual smash product $\_\otimes\_$ on domains and the guaranteed isomorphism is $\tau:A_\perp\otimes B_\perp\cong(A\times B)_\perp$. If $F=\_\times B$, then $\widetilde{F}$ is a nonstandard smash product where maps $\widetilde{F}(A)\to C$ correspond to left strict maps $A\times B\to C$ (see [23]).

**Example 4.13** Let $\mathsf{C}$ again be the category $\mathsf{BDOM}$ of (possibly) bottomless Scott domains with $H$ the "lift" monad, $(\ )_\perp$ and let $\mathsf{D}$ be the category $\mathsf{pBDOM}$ of (possibly) bottomless Scott domains with partial maps and identity monad. Then the inclusion $\mathsf{BDOM}\to\mathsf{pBDOM}$ trivially lifts. The right adjoint $G^*$ of Theorem 4.11 is now exactly the standard equivalence between $\mathsf{pBDOM}$ and $\mathsf{SDOM}$.

**Example 4.14** The Semantics of Weakening and Contraction: Let $(\mathsf{C},\otimes,I)$ be a symmetric monoidal category with strong, commutative monad $H$. If $\mathsf{C}^H$ has coequalizers of reflexive pairs, and $F=\_\otimes\_$ with corresponding monads $H\otimes H$ and $H$ respectively, then $\widetilde{F}$ produces universal bimorphisms, i.e. $\widetilde{F}(A,B)=A\otimes^H B$, in the notation of Jacobs [6]. $\mu_X\otimes^H\mu_Y\cong\mu_{X\otimes Y}$. $\mu_I$ is the neutral element of $\otimes^H$ and $\mathsf{C}^H$ is a symmetric monoidal category(*smc*) where the free functor preserves the *smc* structure [18].

**Example 4.15** If $\mathsf{C}$ is a *ccc* with strong commutative monad $H$, and $C^H$ has coequalizers of reflexive pairs, then $\mathsf{C}^H$ is an exponential ideal and $(\mathsf{C}^H)_{\overline{H}}$ is a *ccc*. Thus we arrive at a *ccc* whose objects are *E-M* algebras.

**Example 4.16** Let $\mathsf{C} = \mathsf{SET}$ with strong, commutative monad $P$, the power set functor. The product bifunctor $F$ lifts to $\mathsf{SET}^P$ = the category of complete semilattices. $\widetilde{F}(A,B) = A \otimes B$ exists and $P(A) \otimes P(B) \cong P(A \times B)$. The associated *ccc* category is complete semilattices and arbitrary set functions. The monad $P^+$, the non-empty power set functor, generates affine complete semilattices [6].

Many other applications of these lifting results are possible. For example the above lifting results can be used to demonstrate that starting with a partial cartesian category with partial maps and a few additional assumptions one can always generate a *ccc* of total maps. This is a systematic process. Thus for example the functorial relationships that exist between Plotkin's partial map semantics and Scott's total semantics is a derivable phenomena rather than simply an observable one. In short, Scott semantics must arise from Plotkin's. Details of these observations can be found in [18] and [19].

## References


[1] Brookes S., and Geva S., *Computational comonads and intensional semantics*, Applications of Category Theory, LMS LNS **177**, (1992), Cambridge University Press, pp. 1-44.

[2] Barr M., and Wells C., "Category Theory for Computing Science," Prentice Hall International, 1990.

[3] Freyd P., *Algebraically complete categories*, preprint, 1991.

[4] Freyd P., and Mulry P., and Rosolini G., and Scott D., *Extensional PERs*, Information and Computation, **98** (1992), pp. 211-227.

[5] Hyland H.M.E., *The effective topos*, L. E. J. Brouwer, Centenary Symposium, North Holland, 1982.

[6] Jacobs B., *Semantics of weakening and contraction*, Annals of Pure and Applied Logic, **69** (1994), pp. 73-106.

[7] Johnstone P. T., *Adjoint lifting theorems for categories of algebras*, Bull. London Math. Soc., **7** (1975), pp. 294-297.

[8] Kock A., *Strong functors and monoidal monads*, Arch. Math., **23** (1972), pp. 113-120.

[9] Liang S., and Hudak P., and Jones M., *Monad transformers and modular interpreters*, Proceedings of POPL'95, (1995), pp. 333-343.

[10] Manes E. G., *A Triple Miscellany: Some Aspects of the Theory of Algebras Over a Triple*, PhD Thesis, Wesleyan University, 1967.

[11] Moggi E.,*The Partial Lambda-Calculus*, PhD Thesis, University of Edinburgh, 1988.

[12] Moggi E., *An Abstract View of programming languages*, Technical Report ECS-LFCS-90-113, LFCS, University of Edinburgh, 1990.

[13] Moggi E., *Notions of computations and monads*, Information and Computation **93** (1991), 55-92.

[14] Mulry P. S., *Generalized Banach-Mazur functionals in the topos of recursive sets*, Journal of Pure and Applied Algebra **26** (1982), 71-83.

[15] Mulry P. S., *Strong monads, algebras and fixed points,* Applications of Category Theory, LMS LNS **177** (1992), Cambridge University Press, pp. 202-216.

[16] Mulry P. S., *Lifting theorems for Kleisli categories*, Lecture Notes in Computer Science **802** (1994), pp. 304-319.

[17] Mulry P. S., *Partial map classifiers and partial cartesian closed categories* Theoretical Computer Science **136** (1994), pp. 109-123.

[18] Mulry P. S., *Lifting results for categories of algebras*, Theoretical Computer Science, to appear.

[19] Mulry P. S., *Algebras and monads in semantics*, preprint.

[20] Plotkin G., *Denotational semantics with partial functions*, Lecture Notes for CSLI, 1985.

[21] Rosolini G., *Continuity and effectiveness in topoi*, Ph.D.thesis, Oxford, 1986.

[22] Seely R., *Linear logic and coalgebras*, Categories in Computer Science and Logic **92** (1989), 371-382, AMS.

[23] Simpson, A. *Recursive types in Kleisli categories*, preprint.

[24] Tannen, V. *A calculus for collections and aggregates*, Lecture Notes in Computer Science **1290** (1997).

[25] Wadler, P. *Comprehending monads*, Mathematical Structures in Computer Science **2** (1992), 461-493, Cambridge University Press.