

# Coalgebraic Monads

Neil Ghani <sup>a,1,4</sup> Christoph Lüth <sup>b,2</sup> Federico De Marchi <sup>a,3,4</sup>

<sup>a</sup> *Department of Maths and Computer Science, University of Leicester, England*

<sup>b</sup> *FB 3 – Mathematik und Informatik, Universität Bremen*

---

## Abstract

This paper introduces coalgebraic monads as a unified model of term algebras covering fundamental examples such as initial algebras, final coalgebras, rational terms and term graphs. We develop a general method for obtaining finitary coalgebraic monads which allows us to generalise the notion of rational term and term graph to categories other than **Set**. As an application we sketch part of the correctness of the the term graph implementation of functional programming languages.

---

## 1 Introduction

Initial algebra semantics has long been regarded as one of the cornerstones of the semantics of programming languages. Within this paradigm, the syntax of a language is modelled as an initial algebra, consisting of the finite terms of the language while the semantics of the language, is given as the unique map from the initial algebra into some other algebra. A similar situation arises in the theory of datatypes, where the initial algebra consists of the finite terms built from the constructors of the datatype while initiality allows functions to be defined over the datatype via structural recursion.

Categorically, one regards such an initial algebra as the initial algebra of an endofunctor  $F$ , which represents the language or datatype. In order to incorporate fundamental features, such as *variables* and *substitution*, into the framework, one considers not a single initial algebra, but rather, for each object  $X$ , the initial  $F$ -algebra *over*  $X$ , ie the initial  $X + F$ -algebra. The mapping sending an object  $X$ , thought of as an object of variables, to (the carrier of) the initial  $X + F$ -algebra defines the free *monad* over  $F$ . The multiplication

---

<sup>1</sup> Email: [ng13@mcs.le.ac.uk](mailto:ng13@mcs.le.ac.uk)

<sup>2</sup> Email: [cx1@informatik.uni-bremen.de](mailto:cx1@informatik.uni-bremen.de)

<sup>3</sup> Email: [fdm2@mcs.le.ac.uk](mailto:fdm2@mcs.le.ac.uk)

<sup>4</sup> Research supported by EPSRC grant GRM96230/01: *Categorical Rewriting: Monads and Modularity*

of the monad provides an abstract representation of substitution, the unit models the variables, while the freeness of the monad models the inductive nature of the initial algebra. Applications to base categories other than **Set** have proved fruitful in many situations, e.g. the study of  $S$ -sorted algebraic theories as monads over  $\mathbf{Set}^S$ , the study of categories with structure using monads over **Graph** or **Cat** [8], the study of rewriting using monads over **Pre** or **Cat** [14] and the study of higher order abstract syntax using monads over  $\mathbf{Set}^{\mathbb{F}}$  [10] (where  $\mathbb{F}$  is the skeleton of the category of finite sets).

Of course, there are other term algebras other than the initial algebra of finite terms. For example, instead of finite terms, what about infinite terms? From the computational perspective one may argue that one should consider only those infinite terms which are computable in some sense. Or, from a recursion-theoretic perspective, what about rational terms which are those terms definable by recursive equations of a certain form? From the perspective of the implementation of functional languages, what about term graphs, which are the dominant model of syntax within that field. These questions are of clear practical importance and the papers cited above suggest the generalisation of these term algebras beyond the category of **Set**.

There has already been some work within the coalgebras community within this direction. Indeed, Moss [15] gave what amounts to a rather full answer in the case of the term algebra of finite and infinite terms by showing that they form the final  $X + F$ -coalgebra, thereby elegantly dualising the the initial algebra characterisation of finite syntax trees. In addition, he showed that the collection of these coalgebras also forms a monad and thereby meeting our requirement for substitution to be taken into account. These results were independently discovered in [1] and in [11] although the latter paper uses a more restrictive setting. Another term algebra, namely the rational terms, has been considered recently [2] but the authors comment on the difficulties of generalising their work beyond the category **Set**. Nevertheless, these results deal only with the specific term algebras and we don't want to tackle each term algebra on a case-by-case basis. This paper introduces *coalgebraic monads* as a uniform framework for term algebra. In detail, we

- Introduce the notion of a coalgebraic monad and demonstrate that this definition captures a number of important examples.
- Provide a general theorem for building monads from coalgebras, and use this theorem to prove that the key examples of rational terms and term graphs are coalgebraic monads.
- Demonstrate that coalgebraic monads have good properties by showing how one can solve equations over them. This allows us to prove part of the correctness of the implementation of functional programming languages via term graphs.

We feel that the term graph monad is possibly the most important contribution of this work. While the denotational semantics of functional languages

provides an elegant framework for reasoning, the semantics of the implementation of functional languages has remained relatively low level. While some attempts to model term graphs using abstract techniques have been made [6], they have yet to make a significant impact within the functional programming community. We hope the naturalness and simplicity of the coalgebraic model of term graphs, for example demonstrated by our work on recursion in coalgebraic monads, will help to bridge this gap.

The paper is structured as follows. We introduce the notion of a coalgebraic monad in Section 2. Section 3 contains a general theorem for proving that notion of syntax is a monad and applies this to the cases of the rational monad and the term graph monad. Section 4 extends Section 3 to cover coalgebraic monads while Section 5 contains our partial correctness result.

## 2 Coalgebraic Monads

Let  $F : \mathcal{C} \longrightarrow \mathcal{C}$  be a functor, which we may think of as arising from a signature of some form. If  $TX$  is some set of terms built from  $F$  using a set of variables  $X$ , then we take the following properties as desirable

- $TX$  should contain all the variables  $X$  and be closed under applications of term constructors from  $F$ . Thus  $TX$  should be an  $X + F$ -algebra.
- Every term  $t \in TX$  should either be a variable or start with a term constructor from  $F$ . Thus  $TX$  should carry an  $X + F$ -coalgebra structure, which ought to be the inverse of the algebra map.
- In order to have a well behaved notion of substitution, the map sending  $X$  to  $TX$  should be a monad

Thus a monad will be  $F$ -coalgebraic iff for each  $X$ ,  $TX$  is an  $X + F$ -fixed point or, more abstractly,  $T$  is a  $(\text{Id} + F \circ -) : [\mathcal{C}, \mathcal{C}] \longrightarrow [\mathcal{C}, \mathcal{C}]$  fixed point<sup>5</sup>. Although this is the intuition underlying a coalgebraic monad, we formally introduce the notion by first noting:

**Lemma 2.1** *Let  $(T, \eta, \mu)$  be a monad and  $F$  an endofunctor on a category  $\mathcal{C}$ . There is a bijection between natural transformations  $\tau : F \longrightarrow T$  and natural transformations  $\alpha : FT \longrightarrow T$  making the following diagram commute*

$$(1) \quad \begin{array}{ccc} FTT & \xrightarrow{F\mu} & FT \\ \alpha_T \downarrow & & \downarrow \alpha \\ TT & \xrightarrow{\mu} & T. \end{array}$$

<sup>5</sup> we shall ignore the size issues here since they can easily be dealt with

**Proof.** Given  $\tau$ , define  $\alpha = \mu.\tau T$ . Commutation of (1) follows immediately by naturality of  $\tau$  and the associativity of  $\mu$ :

$$\begin{array}{ccccc}
 FTT & \xrightarrow{\tau TT} & TTT & \xrightarrow{\mu T} & TT \\
 F\mu \downarrow & & \downarrow T\mu & & \downarrow \mu \\
 FT & \xrightarrow{\tau T} & TT & \xrightarrow{\mu} & T.
 \end{array}$$

Conversely, given  $\alpha$ , define  $\tau = \alpha.F\eta$ . The two mappings are easily shown to be mutually inverse.  $\square$

Condition (1) says just that, if we think of  $\alpha$  as transforming  $F$  terms over  $T$  into  $T$  terms, it doesn't make any difference if we multiply two terms under the  $F$  context and then transform, or rather transform the upper term and then multiply it with the second. In other words, (1) implies that  $\mu : \alpha T \longrightarrow \alpha$  is an  $F$ -algebra homomorphism. The presence of such a structure on a monad  $T$ , gives rise to another monad as can easily be verified by diagram chasing. We shall henceforth assume  $\mathcal{C}$  to have all finite coproducts. Also, given a diagram  $D$  with colimit  $X$ , we shall indicate with  $\bar{d}$  the colimiting map  $\bar{d} : d \longrightarrow X$  for any object  $d$  in  $D$ .

**Lemma 2.2** *Let  $(T, \eta, \mu)$  be a monad on  $\mathcal{C}$  and  $F$  an endofunctor on  $\mathcal{C}$ . Let  $\alpha : FT \longrightarrow T$  be a natural transformation such that (1) commutes. Define  $\bar{\eta} = \text{inl} : \text{Id} \longrightarrow \text{Id} + FT$ ; and*

$$\bar{\mu} : \text{Id} + FT + FT(\text{Id} + FT) \xrightarrow{[\text{Id} + FT + FT[\eta, \alpha]]} \text{Id} + FT + FT^2 \xrightarrow{[\text{Id} + FT, F\mu]} \text{Id} + FT.$$

*Then,  $(\text{Id} + FT, \bar{\eta}, \bar{\mu})$  is a monad, and  $[\eta, \alpha] : \text{Id} + FT \longrightarrow T$  is a monad morphism.*

We can now define a coalgebraic monad and then provide some examples.

**Definition 2.3** Let  $F$  be an endofunctor on a category  $\mathcal{C}$ . An  $F$ -coalgebraic monad on  $\mathcal{C}$  is a 4-tuple  $(T, \eta, \mu, \tau)$  such that  $(T, \eta, \mu)$  is a monad on  $\mathcal{C}$  and  $\tau$  is a natural transformation between  $F$  and  $T$  for which the monad morphism  $[\eta, \mu.\tau T] : \text{Id} + FT \longrightarrow T$  is an isomorphism.

A morphism of  $F$ -coalgebraic monads between  $(T, \eta, \mu, \tau)$  and  $(T', \eta', \mu', \tau')$  is a monad morphism  $\phi$  between  $T$  and  $T'$  such that  $\phi\tau = \tau'$ .

**Proposition 2.4 (Initial Coalgebraic Monad)** *Let  $(T^\mu, \eta, \mu)$  be the free monad over an endofunctor  $F : \mathcal{C} \longrightarrow \mathcal{C}$ . Then,  $T^\mu$  is the initial  $F$ -coalgebraic monad.*

**Proof.** Freeness gives a natural transformation  $\tau : F \longrightarrow T^\mu$ . Verifying the conditions of lemma 2.2 shows that  $\phi = [\eta, \mu.\tau T] : 1 + FT^\mu \longrightarrow T^\mu$  is a monad morphism. Next, the natural transformation  $\text{inr} \circ F\eta : F \longrightarrow \text{Id} + FT^\mu$  induces, by freeness of  $T^\mu$ , a monad morphism  $\psi : T^\mu \longrightarrow 1 + FT^\mu$ . That  $\phi$

and  $\psi$  are mutually inverse are diagram chases, which proves that  $T^\mu$  is an  $F$ -coalgebraic monad. Given any other  $F$ -coalgebraic monad  $(T', \tau')$ , freeness and the transformation  $\tau' : F \longrightarrow T'$  gives a unique monad morphism  $! : T^\mu \longrightarrow T'$  such that  $\tau' = !\tau$ .  $\square$

A slicker proof is possible when, for each object  $X$ ,  $T^\mu(X)$  is the initial  $X + F$ -algebra. In such a setting,  $T^\mu$  is the initial algebra of the endofunctor  $(\text{Id} + F \circ -) : [\mathcal{C}, \mathcal{C}] \longrightarrow [\mathcal{C}, \mathcal{C}]$  and since all initial algebras are isomorphisms we get  $T^\mu$  is isomorphic to  $1 + FT^\mu$ . Initiality follows since every  $F$ -coalgebraic monad is a  $(\text{Id} + F \circ -)$ -algebra.

**Lemma 2.5 (Final Coalgebraic Monad)** *Let  $F : \mathcal{C} \longrightarrow \mathcal{C}$  be and let  $T^\nu$  be the final  $(\text{Id} + F \circ -)$ -coalgebra. Then  $T^\nu$  is the final  $F$ -coalgebraic monad.*

**Proof.** That  $T^\nu$  is a monad such that there is an isomorphism  $[\eta, \alpha] : \text{Id} + FT^\nu \longrightarrow T^\nu$  is proved in [1]. Furthermore, that  $\alpha$  satisfies condition (1) also follows from their substitution theorem. Thus  $T^\nu$  is  $F$ -coalgebraic. Any other  $F$ -coalgebraic monad  $S$  is an  $(\text{Id} + F \circ -)$ -coalgebra and hence there is a unique  $(\text{Id} + F \circ -)$ -coalgebra homomorphism between  $S$  and  $T^\nu$ . This is also a monad morphism, as one can prove with a bit of diagram chasing, using the finality of  $T^\nu$ . Uniqueness follows from finality.  $\square$

A number of coalgebraic monads over **Set** arise as subsets of infinite terms over a signature. When working over **Set** and functors arising from signatures, one can prove a set of infinite terms to be a coalgebraic monad by equipping them with a notion of substitution which is the restriction of that of  $T^\nu$ . This way, one can show that the following are all coalgebraic monads over **Set**: i) infinite terms which contain only a finite number of variables; ii) locally finite terms [7], i.e. finite and infinite terms which have the property that from every node, there is a finite path to a leaf; iii) rational terms are terms with a finite number of subterms, or, more formally, the free iterative theory over a signature [9]. However, the notion of a coalgebraic monad also captures other syntactic structures, such as term graphs, which are used in the implementation of functional programming languages to model recursion and sharing via use of cycles and multiple edges. They can be thought of as labelled graphs allowing cycles and multiple edges.

We now develop a general theorem for deriving monads as pointwise colimits and use this result to define the rational and term graph monads.

### 3 Monads as Pointwise colimits

Inherent in the notions of signature, terms, substitution etc, is the concept of *arity* which categorically means the representing monad has a rank. To understand this condition, the initial algebra  $T_\Sigma X$  built over a signature satisfies

$$(2) \quad T_{\Sigma}(X) = \bigcup_{X_0 \subset X \text{ is finite}} T_{\Sigma}(X_0)$$

This equation holds because all the operators in  $\Sigma$  have a finite arity and thus a term built over  $X$  can only contain a finite number of variables. Such monads are *finitary* and we restrict our attention to them as they capture most of the key examples. The relationship between signatures and their representing monads can be generalised to *lfp-categories* [3], i.e. cocomplete categories generated by a set of *finitely presentable* objects, where an object is finitely presentable if its covariant hom functor preserves filtered colimit. If  $\mathcal{C}$  is an lfp-category, let  $\mathcal{C}_{\text{fp}}$  be the full subcategory of finitely presentable objects with inclusion  $J : \mathcal{C}_{\text{fp}} \longrightarrow \mathcal{C}$ . A functor  $F : \mathcal{C} \longrightarrow \mathcal{D}$  is *finitary* if it preserves filtered colimits, or equivalently if it is isomorphic to the left Kan extension along  $J$  of its restriction to  $\mathcal{C}_{\text{fp}}$ . Hence, the category  $\text{Fin}[\mathcal{C}, \mathcal{D}]$  of finitary functors from  $\mathcal{C}$  to  $\mathcal{D}$  is equivalent to the functor category  $[\mathcal{C}_{\text{fp}}, \mathcal{D}]$ .

In the case of  $\mathcal{D} = \mathcal{C}$ , the category of finitary endofunctors is monoidal with unit the identity and multiplication given by composition. Similarly  $[\mathcal{C}_{\text{fp}}, \mathcal{C}]$  is also monoidal with unit the inclusion  $J$  and with multiplication given by  $F \diamond G = \text{Lan}_J F \circ G$ . Now to give a finitary monad on  $\mathcal{C}$  is to give a monoid in  $\text{Fin}[\mathcal{C}, \mathcal{C}]$  which is thus a monoid in the equivalent category  $[\mathcal{C}_{\text{fp}}, \mathcal{C}]$  [13]. Thus we can prove that rational terms, term graphs or some other term algebra form a finitary monad by proving their restriction to  $[\mathcal{C}_{\text{fp}}, \mathcal{C}]$  is a monoid and use the left Kan extension to get a monad. Such a monoid can be obtained from what we call a *Kleisli monoid*.

**Definition 3.1** A *Kleisli monoid* for an lfp category  $\mathcal{C}$  consists of

- a function  $T$  assigning to each object  $X$  in  $\mathcal{C}_{\text{fp}}$  an object  $TX$  in  $\mathcal{C}$ ;
  - for each  $X$  in  $\mathcal{C}_{\text{fp}}$  a map  $\eta_X : X \longrightarrow TX$  in  $\mathcal{C}$ ;
  - for objects  $X, Y$  in  $\mathcal{C}_{\text{fp}}$ , a lifting function  $s_{X,Y} : \mathcal{C}(X, TY) \longrightarrow \mathcal{C}(TX, TY)$ ;
- satisfying the following conditions:

$$s_{X,X}(\eta_X) = 1_{TX} \quad s_{X,Y}(f) \cdot \eta_X = f \quad s_{X,Z}(s_{Y,Z}(g) \cdot f) = s_{Y,Z}(g) \cdot s_{X,Y}(f).$$

In the rest of the paper, we shall omit the subscript to the function  $s$ , whenever possible. The definition doesn't substantially differ from that of a Kleisli triple for a category  $\mathcal{C}$  and, not surprisingly, defines a monoid in the category  $[\mathcal{C}_{\text{fp}}, \mathcal{C}]$  and hence a monad on  $\mathcal{C}$ .

**Lemma 3.2** *If  $\mathcal{C}$  is an lfp category, then every Kleisli monoid for  $\mathcal{C}$  defines a monoid in the category  $[\mathcal{C}_{\text{fp}}, \mathcal{C}]$ .*

**Proof.** Firstly,  $T$  extends to a functor  $T : \mathcal{C}_{\text{fp}} \longrightarrow \mathcal{C}$  and a natural transformation  $\eta : J \longrightarrow T$  by setting  $T(f) = s(\eta \cdot f)$ . Multiplication requires maps

$\mu_S : (T \square T)S \longrightarrow TS$ . Since  $(T \square T)S$  is defined via the Kan extension

$$(T \square T)S = \mathbf{Lan}_J T(TS) = \int_{n \in \mathcal{C}_{\text{fp}}} \mathcal{C}(n, TS) \otimes Tn \longrightarrow TS$$

we therefore require a wedge  $w_n : \mathcal{C}(n, TS) \otimes Tn \longrightarrow TS$  which, via the universal property of the tensor, is a family of maps  $\mathcal{C}(n, TS) \longrightarrow \mathcal{C}(Tn, TS)$  which is given by the lifting of the Kleisli monoid. That these maps do indeed form a wedge follows from the laws of the Kleisli monoid. Naturality follows from the parametricity theorem for coends while the laws of a monoid again boils down to the laws of the Kleisli monoid.  $\square$

The rational monad and the term graph monad both arise as pointwise colimits. As such, much of the technical reasoning concerning their definition relies on properties of colimits, which we formalise using *Lax slice categories*.

**Definition 3.3** Let  $\mathcal{C}$  be a 2-category and  $X$  an object of  $\mathcal{C}$ . The *lax slice 2-category*  $\mathbf{Lax}_X$  has as objects maps  $f : Y \longrightarrow X$ . Arrows are given by  $\mathbf{Lax}_X(f, g) = \{\langle h, \alpha \rangle \mid \alpha : f \Rightarrow gh\}$ . If  $\alpha : f \longrightarrow g$  and  $\beta : g \longrightarrow h$  are morphisms in  $\mathbf{Lax}_X$ , we write their composite as  $\beta \diamond \alpha : f \longrightarrow h$ . Given maps  $\langle h, \alpha \rangle : f \longrightarrow g$  and  $\langle h', \alpha' \rangle : f \longrightarrow g$ , a 2-cell  $\langle h, \alpha \rangle \longrightarrow \langle h', \alpha' \rangle$  consists of a 2-cell  $\theta : h \longrightarrow h'$  in  $\mathcal{C}$  such that  $\alpha' = g\theta.\alpha$ .

The usual definition of a slice category  $\mathcal{C}/X$  is the lax slice category on the 2-category obtained by adding only the identity 2-cells to  $\mathcal{C}$ . Lax slice categories allow us to state the following:

**Lemma 3.4** *Let  $\mathcal{C}$  be a category and consider the lax slice category  $\mathbf{Lax}_{\mathcal{C}}$  built over the 2-category  $\mathbf{Cat}$ . If  $\langle H, \alpha \rangle \in \mathbf{Lax}_{\mathcal{C}}(F, G)$ , then the family of arrows  $\overline{Hd}.\alpha_d$  defines a cocone over  $F$ , which in turn induces a map in  $\mathcal{C}$   $\text{colim}\alpha : \text{colim}F \longrightarrow \text{colim}G$ . In addition,  $\text{colim}1_F = 1 : \text{colim}F \longrightarrow \text{colim}F$  and  $\text{colim}(\beta \diamond \alpha) = \text{colim}\beta.\text{colim}\alpha$ . Finally, given a 2-cell  $\alpha \longrightarrow \alpha'$ , then  $\text{colim}\alpha = \text{colim}\alpha'$ .*

### 3.1 When is a Pointwise Colimit a Monad?

Let  $\mathcal{C}$  be a category,  $\mathcal{I} : \mathcal{C} \longrightarrow \mathbf{Cat}$  a functor, and  $\mathbf{K}_{\mathcal{C}}$  the constant functor mapping each object to  $\mathcal{C}$ . The rest of this section finds conditions on a natural transformation  $U : \mathcal{I} \Rightarrow \mathbf{K}_{\mathcal{C}} : \mathcal{C} \longrightarrow \mathbf{Cat}$  so that the assignment  $X \mapsto \text{colim}U_X$  defines a finitary monad, denoted  $T$ , on  $\mathcal{C}$ . Intuitively,  $\mathcal{I}$  maps each object  $X$  to the subcategory of  $(X + F)\text{-coalg}$  consisting of those coalgebras we want to capture within our monad. The  $X$ -th component of the natural transformation  $U$  plays the role of the forgetful functor from this subcategory to the base category  $\mathcal{C}$ , and  $\text{colim}U_X$  will then be the object representing the collection we are interested in, i.e. the action of our monad  $T$  on  $X$ .

We prove that  $T$  defined as above is a monad by proving that its restriction to  $\mathcal{C}_{\text{fp}}$  supports a Kleisli monoid structure. Since we have the action  $TX =$

$\text{colim}U_X$  for  $X$  finitely presentable, we turn to our candidate for the unit. If we assume that for each finitely presentable object  $X \in \mathcal{C}_{\text{fp}}$  there is an object  $i_X \in \mathcal{I}X$  such that  $U_X(i_X) = X$ , then we can define  $\eta_X$  to be the inclusion  $\overline{i_X} : X = U_X(i_X) \longrightarrow \text{colim}U_X = TX$ . In our examples,  $i_X$  will be the coalgebra  $\text{inl} : X \longrightarrow X + FX$ , whose inclusion embeds the variables into the colimit.

Finally, we turn to the lifting functions, for which extra properties are needed. Given a map from an object  $X$  to  $TY$  (the object representing the  $Y + F$ -coalgebras), we extend it to a map from  $TX$  to  $TY$  by mapping each  $X + F$ -coalgebra into a  $Y + F$ -coalgebra and then using the universal property of  $\text{colim}U_X = TX$ . This is done by first observing that  $X$  can be actually mapped to  $TY_0$  for some subobject  $Y_0$  (which represents the set of variables on which the terms we are substituting are actually built), then adding  $Y_0$  to the carrier of each  $X + F$ -coalgebra, thus making it into a  $Y + F$  one. Formally, this procedure is captured by what we call a lifting:

**Definition 3.5**  $\mathcal{I}$  is said to have a *lifting*  $L$  when

- For each object  $X \in \mathcal{C}$  and each  $g \in \mathcal{I}X$ , there is an arrow  $\langle L(g), g^L \rangle : U_{Ug} \longrightarrow U_X$  in  $\mathbf{Lax}_{\mathcal{C}}$ , i.e. a functor  $L(g) : \mathcal{I}Ug \longrightarrow \mathcal{I}X$  and a natural transformation  $g^L : U \Rightarrow U.L(g)$
- For each arrow  $k : g \longrightarrow h$  in  $\mathcal{I}X$ , there exists a natural transformation  $k^L : L(g) \longrightarrow L(h).\mathcal{I}Uk$  such that  $h^L.\mathcal{I}Uk = Uk^L.g^L$

$$\begin{array}{ccc}
 \mathcal{I}Ug & \xrightarrow{L(g)} & \mathcal{I}X \\
 U \searrow & \Rightarrow_{g^L} & \swarrow U \\
 & \mathcal{C} & 
 \end{array}
 \qquad
 \begin{array}{ccc}
 \mathcal{I}Ug & \xrightarrow{\mathcal{I}Uk} & \mathcal{I}Uh \\
 L_g \searrow & \Rightarrow_{k^L} & \swarrow L_h \\
 & \mathcal{I}X & 
 \end{array}$$

**Lemma 3.6** *Let  $X$  and  $Y$  be finitely presentable, and suppose that  $\mathcal{I}Y$  is a filtered category. Then, any map  $f : X \longrightarrow \text{colim}U_Y$ , induces a map  $s(f) : \text{colim}U_X \longrightarrow \text{colim}U_Y$ .*

**Proof.** Since  $X$  is finitely presentable and  $\mathcal{I}Y$  is filtered,  $f$  factors as  $f = \overline{i_f}.f^+$  for some  $i_f \in \mathcal{I}Y$ . This defines a functor  $Lf = L(i_f)\mathcal{I}(f^+)$  and also a natural transformation  $f^L = i_f^L \diamond 1$

$$\begin{array}{ccc}
 X & \xrightarrow{f} & \text{colim}U_Y \\
 f^+ \searrow & & \uparrow \overline{i_f} \\
 & & U_Y(i_f)
 \end{array}
 \qquad
 \begin{array}{ccccc}
 \mathcal{I}X & \xrightarrow{\mathcal{I}(f^+)} & \mathcal{I}U(i_f) & \xrightarrow{L_{i_f}} & \mathcal{I}Y \\
 U \searrow & \Rightarrow_1 & \downarrow U & \Rightarrow_{i_f^L} & \swarrow U \\
 & & \mathcal{C} & & 
 \end{array}$$

and hence, by lemma 3.4, the map  $s(f) = \text{colim}f^L : \text{colim}U_X \longrightarrow \text{colim}U_Y$ .

The construction of  $s(f)$  appears to depend upon the factorisation  $f = \overline{i_f}.f^+$ . This is actually not the case. In fact, suppose  $f = \overline{i_h}.h$  is another

factorisation, inducing the functor and natural transformation

$$L(i_h)\mathcal{I}(h) : \mathcal{I}X \longrightarrow \mathcal{I}Y \text{ and } i_h^L \diamond 1 : U \Rightarrow UL(i_h)\mathcal{I}(h)$$

By lemma 3.4, we have a map  $\text{colim}(i_h^L \diamond 1) : TX \longrightarrow TY$ . Because  $\mathcal{I}Y$  is filtered, there exist  $i, k : i_f \longrightarrow i$  and  $k' : i_h \longrightarrow i$  such that  $f$  factorises as  $f = \bar{i}.g$ , for a given map  $g : X \longrightarrow U_Y(i)$ , with  $Uk.f^+ = g = Uk'.h$ . Let's focus on  $k : i_f \longrightarrow i$ . The lifting  $L$  determines  $k^L : L(i_f) \Rightarrow L(i).\mathcal{I}Uk$ . Then

$$i^L \diamond 1 = i^L \mathcal{I}(Uk)\mathcal{I}(f^+) = (Uk^L.i_f^L)\mathcal{I}(f^+) = Uk^L \mathcal{I}(f^+).i_f^L \mathcal{I}(f^+)$$

where the first equality comes from  $\mathcal{I}(g) = \mathcal{I}(Uk)\mathcal{I}(f^+)$ , the second is from the lifting of  $k^L$  and the third is the distribution of horizontal over vertical composition of natural transformations. Thus  $i^L \diamond 1 = Uk^L \mathcal{I}(f^+).i_f^L \diamond 1$ , and by lemma 3.4, we have that  $\text{colim}(i^L \diamond 1) = \text{colim}(i_f^L \diamond 1)$ . Analogously,  $\text{colim}(i^L \diamond 1) = \text{colim}(i_h^L \diamond 1)$ , hence  $s(f)$  doesn't depend on the factorisation.  $\square$

Reusing the notation above, we define for the canonical factorisation  $f = \bar{i}_f.f^+$  the functor  $L(f) = L(i_f).\mathcal{I}(f^+)$  and  $f^L = i_f^L \mathcal{I}(f^+)$ . Having obtained our candidates for a Kleisli triple, we now verify the three laws for which we need further assumptions. For the lifting of the unit, note first that there is a factorisation  $\eta_X = \bar{i}_X.1$ . Thus, by lemma 3.4,  $s(\eta) = \text{colim}(i_X^L \diamond 1) = \text{colim}i_X^L \text{colim}1 = \text{colim}i_X^L$ . This proves that

**Lemma 3.7** *Let  $X \in \mathcal{C}$ . If  $\text{colim}i_X^L = 1$ , then  $s(\eta_X) = 1_{U_X}$ .*

**Lemma 3.8** *Let  $X$  and  $Y$  be finitely presentable and  $f : X \longrightarrow \text{colim}U_Y$ . If there is a map  $k : L(f)(i_X) \longrightarrow i_f$  in  $\mathcal{I}Y$  such that  $Uk.f_{i_X}^L = f^+$ , then  $s(f).\eta_X = f$ .*

**Proof.**  $s(f)$  is determined by a factorisation  $f = \bar{i}_f.f^+$ . Since  $\eta_X$  is the inclusion from  $U_X(i_X)$  to  $\text{colim}U_X$ ,  $s(f).\eta$  is the  $U_X(i_X)$ -th component of the cocone determined by  $f$  as in lemma 3.6, i.e.  $s(f).\eta = \bar{L}(f)(i_X).f_{i_X}^L$ . Thus:

$$\begin{array}{ccccc} X = U_X(i_X) & \xrightarrow{f_{i_X}^L} & U_Y L(f)(i_X) & & \\ & \searrow f^+ & \downarrow Uk & \swarrow L(f)(i_X) & \\ & & U_Y(i_f) & \xrightarrow{\bar{i}_f} & \text{colim}U_Y \end{array}$$

where the left triangle commutes by assumption and the right since it is part of the universal cocone over  $U_Y$ .  $\square$

**Lemma 3.9** *Let  $X, Y$  and  $Z$  be finitely presentable. Consider two maps  $f : X \longrightarrow \text{colim}U_Y$  and  $g : Y \longrightarrow \text{colim}U_Z$ . Assume, for all  $y \in \mathcal{I}Y$ ,*

$$\begin{array}{ccc} \mathcal{I}Uy & \xrightarrow{L(y)} & \mathcal{I}Y \\ \mathcal{I}(g_y^L) \downarrow & & \downarrow L(g) \\ \mathcal{I}U(L(g)y) & \xrightarrow{L(L(g)y)} & \mathcal{I}Z \end{array}$$

and  $g^L \diamond y^L = (L(g)y)^L \diamond 1$ . Then  $s(s(g).f) = s(g).s(f)$

$$\begin{array}{ccc}
 \mathcal{I}Uy & \xrightarrow{L_y} & \mathcal{I}Y \xrightarrow{L_g} & \mathcal{I}Z \\
 & \searrow U & \Downarrow \Rightarrow_{y^L} U & \searrow U \\
 & & \mathcal{C} & \\
 & & \Downarrow \Rightarrow_{g^L} U & \\
 & & & \mathcal{C}
 \end{array}
 \qquad
 \begin{array}{ccc}
 \mathcal{I}Uy & \xrightarrow{\mathcal{I}(f_y^L)} & \mathcal{I}U(L_g(y)) \xrightarrow{L(L_g(y))} & \mathcal{I}Z \\
 & \searrow U & \Downarrow \Rightarrow_1 U & \searrow U \\
 & & \mathcal{C} & \\
 & & \Downarrow \Rightarrow_{(L_g y)^L} U & \\
 & & & \mathcal{C}
 \end{array}$$

**Proof.** The map  $s(g).f$  can be factorised as follows

$$\begin{array}{ccccc}
 X & \xrightarrow{f} & TY & \xrightarrow{s(g)} & TZ \\
 & \searrow f_x & \uparrow \overline{i_f} & & \uparrow \overline{L(g)(i_f)} \\
 & & Ui_f & \xrightarrow{g_{i_f}^L} & U(L(g)(i_f))
 \end{array}$$

where  $f = \overline{i_f}.f^+$  is the factorisation of  $f$  used in the construction of  $s(f)$  and the square commutes by the construction of  $s(g)$ . Thus we have a factorisation of  $s(g).f$  and the functor  $L(s(g).f)$  can be calculated as

$$L(s(g).f) = L(g^L i_f). \mathcal{I}(g_{i_f}^L). \mathcal{I}(f^+) = L(g).L(i_f). \mathcal{I}(f^+) = L(g).L(f)$$

where the third equality is by definition of  $L(f)$ . From the assumptions, the following equality is also derivable  $(L(g)i_f)^L \diamond 1 = g^L \diamond i_f^L : \mathcal{I}U_X(i_f) \longrightarrow \mathcal{I}Z$ . By precomposing with  $\mathcal{I}f^+$ , we get that the two 2-cells associated to  $L(s(g).f)$  and  $L(g).L(f)$  are equal, therefore  $s(s(g).f) = s(g).s(f)$ .  $\square$

Collecting all the results so far, we get the following:

**Theorem 3.10** *Let  $\mathcal{C}$  be an lfp category,  $\mathcal{I} : \mathcal{C}_{\text{fp}} \longrightarrow \text{Cat}$  a functor such that for every  $X$  in  $\mathcal{C}_{\text{fp}}$ ,  $\mathcal{I}X$  is filtered. Let  $U : \mathcal{I} \Rightarrow \mathbf{K}_{\mathcal{C}} : \mathcal{C}_{\text{fp}} \longrightarrow \text{Cat}$  be a natural transformation with a lifting  $L$ . For arbitrary objects  $X, Y$  and  $Z$  in  $\mathcal{C}_{\text{fp}}$  and maps  $f : X \longrightarrow \text{colim}U_Y$  and  $g : Y \longrightarrow \text{colim}U_Z$  in  $\mathcal{C}$ , assume that*

- *there is an object  $i_X$  in  $\mathcal{I}X$  such that  $U_X(i_X) = X$  and  $\text{colim}i_X^L = 1$ ;*
- *there is a map  $k : L(f)(i_X) \longrightarrow i_f$  such that  $Uk.f_{i_X}^L = f^+$ ;*
- *for all  $y$  in  $\mathcal{I}Y$ ,  $g^L \diamond y^L = (L(g)y)^L \diamond 1$  and*

$$\begin{array}{ccc}
 \mathcal{I}Uy & \xrightarrow{L(y)} & \mathcal{I}Y \\
 \mathcal{I}(g_y^L) \downarrow & & \downarrow L(g) \\
 \mathcal{I}U(L(g)y) & \xrightarrow{L(L(g)y)} & \mathcal{I}Z.
 \end{array}$$

*Then the assignment  $TX = \text{colim}U_X$  carries a Kleisli monoid structure.*

**Corollary 3.11** *Under the assumptions of the previous theorem, the left Kan extension of  $T$  under the inclusion of  $\mathcal{C}_{\text{fp}}$  into  $\mathcal{C}$  is a finitary monad on  $\mathcal{C}$ .*

## 4 Rational and Term Graph Monads

We now use our general theorem to prove that term graphs and rational terms form monads. First we state some general properties of categories of coalgebras. We write  $U_X : (X+F)\text{-coalg} \longrightarrow \mathcal{C}$  for the forgetful functor sending each coalgebra to its carrier.  $U_X$  creates colimits and since  $\mathcal{C}$  is lfp, and hence cocomplete, any functor  $J : D \longrightarrow (X+F)\text{-coalg}$  from a small category  $D$  has a colimit and  $U_X \text{colim} J = \text{colim} U_X J$ . Next,

**Lemma 4.1** *Let  $F : \mathcal{C} \longrightarrow \mathcal{C}$  be a functor. The assignment sending an object  $X$  of  $\mathcal{C}$  to the category  $(X+F)\text{-coalg}$  defines a functor  $\Phi : \mathcal{C} \longrightarrow \mathbf{Cat}$ . In addition,  $U : \Phi \Rightarrow \mathbf{K}_{\mathcal{C}}$  is a natural transformation.*

**Proof.** If  $f : X \longrightarrow Y$  is a map in  $\mathcal{C}$ ,  $\Phi(f)$  sends an  $X+F$ -coalgebra  $(S, h)$  to the  $X+F$ -coalgebra  $(S, (f+1).h)$ . The action of  $\Phi(f)$  on coalgebra morphisms and functoriality are easily proved. Naturality holds since  $U_X = U_Y \Phi(f)$ .  $\square$

In general, we shall instantiate theorem 3.10 with functors which are variously related to  $\Phi$  and whose properties follow from those of  $\Phi$ . Similarly, we define a lifting for  $\Phi$  from which liftings for other functors can be obtained.

**Lemma 4.2** *Let  $(S, g)$  be an  $X+F$ -coalgebra. The map sending any  $S+F$ -coalgebra  $(A, m)$  to the following  $X+F$ -coalgebra defines a lifting for  $\Phi$ .*

$$A + S \xrightarrow{[m, \text{inl}]} S + FA \xrightarrow{g+1} X + FS + FA \xrightarrow{1+[F\text{inr}, F\text{inl}]} X + F(A + S)$$

**Proof.** We have defined the object part of a functor  $L(g) : \Phi U g \longrightarrow \Phi X$  and the action of  $L(g)$  on morphisms is easily defined. The  $(A, m)$ -th component of  $g^L : U_S \longrightarrow U_X.L(g)$  is the inclusion  $g_m^L = \text{inl} : A \longrightarrow A + S$ . Naturality of  $g^L$  is the naturality of  $\text{inl}$ . Given any  $X+F$ -coalgebra map  $k : (S, g) \longrightarrow (S', h)$ , define  $k^L$  to be the natural transformation whose component on an  $(S+F)$ -coalgebra  $m : A \longrightarrow S + FA$  is the following  $X + F$ -coalgebra morphism

$$\begin{array}{ccc} A + S & \xrightarrow{L(g)(m)} & X + F(A + S) \\ \downarrow 1+k & & \downarrow 1+F(1+Uk) \\ A + S' & \xrightarrow{L(h)(\Phi k(m))} & X + F(A + S') \end{array}$$

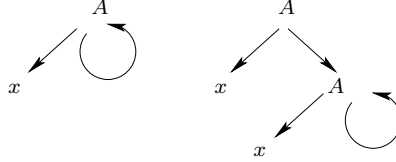
The components of the natural transformations representing the 2-cell condition are  $(1+k).\text{inl} : A \longrightarrow A + S'$  and  $\text{inl} : A \longrightarrow A + S'$ , hence clearly equal.  $\square$

### 4.1 The Rational Monad

Given a signature  $\Sigma$ , rational terms are the set of finite and infinite terms which arise as solutions of systems of equations of the form  $e_1 = t_1, \dots, e_n = t_n$  where each  $t_i$  is a term built from the signature  $\Sigma$  and whose variables are from the union of a fixed set  $X$  with the set  $E = \{e_1, \dots, e_n\}$ . In order to get a solution one insists that each term  $t_i$  is not an element of  $E$ . Rephrasing this

categorically, a rational equation is a function  $\phi : E \longrightarrow X + F_\Sigma T_\Sigma(X + E)$  where  $F_\Sigma$  is the polynomial endofunctor associated to the signature and  $T_\Sigma$  is the initial  $(X + E) + F$ -algebra, as in (2). As we shall see in section 6, all such equations have unique solutions in  $T^\nu(X)$ . For example, the equation  $e = \mathbf{A}(e)$  has as its solution the infinite term  $\mathbf{A}(\mathbf{A}(\dots))$ .

We can think of an equation of the form  $e = t(e, \vec{x})$  as a finite tree with variables  $\vec{x}$  and  $e$  pointing to the root of the system. For example, the equations  $e = \mathbf{A}(x, e)$  and  $e = \mathbf{A}(x, \mathbf{A}(x, e))$  can be represented as follows



Such trees are  $X + F_\Sigma$ -coalgebras on a finite set whose states are the nodes in the graph and whose structure map sends each state to either the term constructor labelling it and the child states, or to the variable labelling the node. The rational monad is thus the colimit of the inclusion of the full subcategory of  $X + F$ -coalgebras with finitely presentable carrier. Taking the full subcategory includes all coalgebra morphisms and hence quotients by bisimulation. This ensures the rational equations above have the same solution. Formally, we take  $\mathcal{I}_R : \mathcal{C} \longrightarrow \mathbf{Cat}$  to be the functor mapping an object  $X$  to the full subcategory of  $X + F$ -coalgebras whose underlying object is finitely presentable. The forgetful functor  $U_X : \mathcal{I}_R(X) \longrightarrow \mathcal{C}$  provides the functor whose pointwise colimit we shall take. We now use theorem 3.10 to prove rational terms form a monad.

**Proposition 4.3** *The assignment  $X \mapsto RX = \text{colim} U_X$  defines a monad.*

**S**ince this functor is finitary, we show that it is a monad by using corollary 3.11 on its restriction to  $\mathcal{C}_{\text{fp}}$ . Functoriality of  $\mathcal{I}_R$  is inherited from that of  $\Phi$  as introduced in lemma 4.1. Similarly, a lifting on  $\mathcal{I}_R$  comes as a restriction of the lifting  $L$  for  $\Phi$ . Crucially for this, the coproduct of finitely presentable objects is finitely presentable which means that  $L(f)$ , when applied to a coalgebra on a finitely presentable object, returns a coalgebra on a finitely presentable object. Since we use the full subcategory, all the components of the required natural transformations in the lifting  $L$  are still available. Finally, all the equational properties of the lifting still hold, hence we have a lifting for  $\mathcal{I}_R$ .

In order to apply corollary 3.11, we still need to verify the three conditions expressed in our main theorem 3.10. The existence of a natural transformation  $\alpha = \text{inl} : \text{Id}_{\mathcal{I}_R} \Rightarrow i_X^L$  ensures, by lemma 3.4, that  $\text{colim} i_X^L = \text{colim} \text{Id} = 1$ .

Now, given a map  $f : X \longrightarrow RY$  in  $\mathcal{C}$ , where  $X$  and  $Y$  are finitely presentable, the coalgebra  $L(f)(i_X)$  is defined by the composite

$$X + Y_0 \xrightarrow{\text{inl}[f^+, Y_0]} Y_0 + FX \xrightarrow{i_f + FX} Y + FY_0 + FX \xrightarrow{Y + [F\text{inl}, F\text{inr}]} Y + F(Y_0 + X)$$

where  $f$  factors as  $f = \overline{i_f}f^+$ . If we take the map  $k : L(f)(i_X) \longrightarrow i_f$  to be  $[f^+, Y_0] : X + Y_0 \longrightarrow Y_0$ , it is now clear that  $Uk.f_{i_X}^L = f^+$ .

Finally, let  $Z$  be finitely presentable,  $g : Y \longrightarrow RZ$  be a map in  $\mathcal{C}$ , and  $y : Y_0 \longrightarrow Y + FY_0$  be a  $Y + F$ -coalgebra. Then, for any  $Y_0 + F$ -coalgebra  $\alpha : A \longrightarrow Y_0 + FA$  in  $\mathcal{I}(Uy)$ , the actions of the functors  $L(g)L(y)$  and  $L(L(g)y)\mathcal{I}(g_y^L)$  on  $\alpha$  determine the same  $Z + F$ -coalgebra

$$\begin{array}{c}
A + Y_0 + Z_0 \\
\downarrow [\alpha, Y_0, Z_0] \\
Y_0 + Z_0 + FA \\
\downarrow (Y+Z_0+[Finl, Finr])[y, Z_0, FA] \\
Y + Z_0 + F(A + Y_0) \\
\downarrow (Z+[Finl, Finr])[i_g[g^+, Z_0], F(A+Y_0)] \\
Z + F(A + Y_0 + Z_0)
\end{array}$$

where  $g$  factors as  $g = \overline{i_g}g^+$  with  $U(i_g) = Z_0$ , and we have omitted the obvious inclusions in the sums and deliberately rearranged the summands. The two corresponding components of the natural transformations  $g^L \diamond y^L$  and  $(L(g)y)^L \diamond \text{Id}$  turn out to be the inclusion  $A \longrightarrow A + Y_0 + Z_0$ , hence being equal. □

#### 4.2 The Term Graph Monad

Term graphs are generalisations of finite syntax trees which allow i) cycles, as in rational equations, to model recursion, and ii) multiple edges, to model sharing. Sharing is fundamental in obtaining efficiency, as computations in a shared subterm need only be performed once, rather than one time for each occurrence. Traditionally, term graphs are defined by labelled graphs [4], but we feel the definition of a term graph as an  $X + F$ -coalgebra is considerably cleaner, as the arity information made explicit in the usual definition is hidden inside the structure map of the coalgebra, and is usually automatically verified when working with coalgebras. We also choose to impose on our term graphs the maximal possible sharing, i.e. all the variables are forced to be called by at most one state. These terms are very close to the ones studied by Hasegawa in his phd thesis [12], although there seem to be some differences. Set theoretically, we define them as follows.

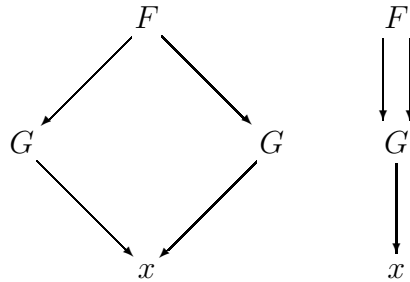
**Definition 4.4** Let  $\Sigma$  be a finitary signature, and  $X$  a set. A *term graph* with variables in  $X$  is a 5-tuple  $(S, V, L, A, r)$  where

- $S$  is a finite set;

- $V$  is a finite subset of  $X$ ;
- $L$  is a function from  $S$  to  $\Sigma$ ;
- $A$  is a function from  $S$  to  $C^*$  such that the length of  $A(v)$  is equal to the arity of  $L(v)$ , where  $C = S + V$ ;
- $r$  is an element of  $C$  such that for any other state  $s$  in  $C$  there is a finite sequence  $a_1, \dots, a_n$  such that  $a_1 = r$ ,  $a_n = s$  and  $a_i$  is an element in  $A(a_{i-1})$ .

Here  $S$  represents the set of states of our graph. States which represent variables are separated from  $S$  and collated in a set  $V$ , which embeds in  $X$ , thus enforcing that each variables is called at most once.  $L$  is a labelling function, which, for each state in  $S$ , says which label is attached to it. States which represent variables have no label attached to them. Finally,  $A$  maps each state in  $S$  to the string consisting of its successor states (children) after the transition indicated by  $L$ . Hence, the length of the string has to be the same as the arity of the function symbol which labels the state. The element  $r$  is a chosen root of the term, from which all the other states can be reached. We will call  $G(X)$  the set of term graphs with variables in  $X$ .

Such a complicated and syntactical definition rewrites categorically in a much nicer way. Indeed, with the data above we can give  $S + V$  an  $X + F$ -coalgebra structure (with finite carrier) in a very obvious way. The fact that  $V$  maps injectively into  $X$  reflects the fact that variables occur at most once. In order to get the set  $G(X)$  as a colimit, deriving both functoriality and monadicity by using our theorem, the idea is to consider the category of such coalgebras. Actually, some more refinements are needed; namely, we want to allow as much choice as possible in the sharing of subterms. Contrary to what we did before, we now don't want to keep bisimilar terms distinct. For example, we want to distinguish between the two following term graphs



because in the first case  $G$  is not shared. In order to maintain them unequal, we need to remove all those arrows which realise bisimulations between two different terms. On the other hand, we still want to identify two copies of the same term, as well as embedded copies of a subterm. These considerations lead us to allow as maps only inclusions.

In order to formulate these ideas in a precise way, and abstract from the category of sets, we will now introduce a few more assumptions on our base category  $\mathcal{C}$  and then use our main theorem to build a term graph monad. The subcategory  $\mathcal{I}_G X$  of the category of  $X + F$ -coalgebras will have as objects

those coalgebras with a finitely presentable carrier such that each variable occurs at most at one state. In other words, we need to split the carrier of the coalgebra in a sum of two objects, one of which will embed in  $X$ , whereas the remaining states will have a labelled transition to some others. Such a splitting reflects the objects  $V$  and  $S$  in the definition above, and, in order to perform it, we need to work in the context of an *extensive category* [5].

**Definition 4.5** A category  $\mathcal{C}$  with finite coproducts is said *extensive* if it admits pullbacks along injections into a coproduct, and any pair of commuting squares

$$\begin{array}{ccccc} A_1 & \xrightarrow{\alpha_1} & A & \xleftarrow{\alpha_2} & A_2 \\ g_1 \downarrow & & f \downarrow & & \downarrow g_2 \\ B & \xrightarrow{\text{inl}} & B + C & \xleftarrow{\text{inr}} & C \end{array}$$

is a pair of pullbacks if and only if the top row is a coproduct diagram in  $\mathcal{C}$ .

Some properties of extensive categories are proved in [5]. We report here the ones which interest us.

**Proposition 4.6** *Let  $\mathcal{C}$  be an extensive category. Then*

- *the pullback of the two injections in a binary coproduct is the initial object;*
- *injections into coproducts are monic (these two property say that sums are disjoint);*
- *any arrow into an initial object is invertible (i.e. initials are strict);*
- *for any objects  $A, B$  and  $C$  in  $\mathcal{C}$ , the canonical map from  $(A \times B) + (A \times C)$  to  $A \times (B + C)$  is an isomorphism, i.e.  $\mathcal{C}$  is distributive.*

We are now in the position to build our monad by means of the theorem. We shall henceforth consider  $\mathcal{C}$  to be also extensive. Given an  $X + F$ -coalgebra  $\gamma : C \longrightarrow X + FC$ , we can then split  $C$  as the sum of objects which are mapped into  $X$  and those which are mapped into  $FC$  by forming the two pullbacks below.

$$\begin{array}{ccccc} C_V & \xrightarrow{\text{inl}} & C & \xleftarrow{\text{inr}} & C_S \\ \gamma_V \downarrow & \lrcorner & \gamma \downarrow & \lrcorner & \downarrow \gamma_S \\ X & \xrightarrow{\text{inl}} & X + FC & \xleftarrow{\text{inr}} & FC \end{array}$$

Now let  $\mathcal{I}_G(X)$  be the subcategory of  $(X + F)\text{-coalg}$  having as objects all those coalgebras  $\gamma : C \longrightarrow X + FC$  with a finite carrier  $C$  and such that  $\gamma_V$  is a monic in  $\mathcal{C}$ , and as maps all coalgebra morphism whose carrier map is a monic in  $\mathcal{C}$ . Furthermore, let the functor  $U_X$  be the natural restriction of the forgetful functor. It's not hard to see that  $\text{colim} U_X$  gives precisely

the set  $G(X)$  we have described above. A colimiting map from the carrier of a coalgebra in  $\mathcal{I}_G(X)$  to  $G(X)$  simply maps each element  $s$  to the smallest subcoalgebra including  $s$ . This coalgebra is clearly rooted at  $s$  itself. In the rest of this section, we shall often omit the subscript in  $\mathcal{I}_G$  and we shall write  $G(X)$  for the colimit of  $U_X$ .

In order for the substitution to work right, we now need to use a different lifting. Let  $\gamma : C \longrightarrow X + FC$  be an object of  $\mathcal{I}X$ . The functor  $L(\gamma) : \mathcal{I}C \longrightarrow \mathcal{I}X$  acts as follows. Let  $\alpha : A \longrightarrow C + FA$  be an object of  $\mathcal{I}C$ . Then the coalgebra  $L(\gamma)(\alpha)$  is defined by the composite

$$A_S + C \xrightarrow{\alpha_S + \gamma} FA + X + FC \xrightarrow{X+[F\text{inl}, F\text{inr}]} X + F(A + C) \xrightarrow{X+F(A_S + [\alpha_V, C])} X + F(A_S + C)$$

If  $f : (A, \alpha) \longrightarrow (B, \beta)$  is a  $C + F$ -coalgebra morphism, then  $L(\gamma)(f)$  is the coalgebra morphism whose carrier map is  $f_S + C$ , where  $f_S$  is obtained by the universal property of the pullback defining  $\beta_S$  in the expected way. In particular, it follows that

$$\begin{array}{ccc} A_S & \xrightarrow{\text{inl}} & A \\ f_S \downarrow & & \downarrow f \\ B_S & \xrightarrow{\text{inl}} & B \end{array}$$

commutes, and since both  $f$  and  $\text{inl}$  are monic,  $f_S$  has to be monic too. This ensures that  $f_S + C$  is monic, and therefore a map in  $\mathcal{I}X$ .

The component of the natural transformation  $\gamma^L : U_C \Rightarrow U_X.L(\gamma)$  on a coalgebra  $\alpha : A \longrightarrow C + FA$  in  $\mathcal{I}C$  is given by the map  $\gamma_\alpha^L = A_S + \alpha_V : A \longrightarrow A_S + C$ . Naturality is an easy check.

The action of the lifting on a map  $\phi : (C, \gamma) \longrightarrow (D, \delta)$  in  $\mathcal{I}X$  has to be a natural transformation  $\phi^L : L(\gamma) \Rightarrow L(\delta)$ . We define its component on  $\alpha : A \longrightarrow C + FA$  in  $\mathcal{I}C$  as the map  $A_S + \phi : A_S + C \longrightarrow A_S + D$ . Naturality is again easy to check. Furthermore, the equality  $U_X(\phi^L)\gamma^L = \delta^L.\mathcal{I}U\phi$  holds, as the components of the two natural transformations for a coalgebra  $\alpha : A \longrightarrow C + FA$  in  $\mathcal{I}C$  both turn out to be the composite

$$A \xrightarrow{A_S + \alpha_V} A_S + C \xrightarrow{A_S + \phi} A_S + D.$$

This defines the lifting. We now move on to the further properties required by theorem 3.10. First of all, we need to define an object  $i_X$  in  $\mathcal{I}X$  such that  $U_X i_X = X$ . Let's put  $i_X = \text{inl} : X \longrightarrow X + FX$ . This coalgebra is in  $\mathcal{I}X$  because  $(i_X)_V$  is the identity on  $X$ , hence monic. We also need to show that  $\text{colim}_X^L = 1$ . For this, using lemma 3.4, it's enough to show that there is a natural transformation  $\lambda : \text{ld} \Rightarrow L(i_X)$ . We define the  $(C, \gamma)$ -th component of  $\lambda$  to be the map  $C_S + \gamma_V : C_S + C_V = C \longrightarrow C_S + X$ , which is a map in  $\mathcal{I}X$  because  $\gamma_V$  is monic by assumption.

Let's now assume  $f$  to be a map from  $X$  to  $GY$  factorising through the carrier of a coalgebra  $\gamma : C \longrightarrow Y + FC$  in  $\mathcal{I}Y$  as

$$X \xrightarrow{f^+} C \xrightarrow{\bar{\gamma}} GY.$$

Then, we need a map  $k : L(f)(i_X) \longrightarrow C$ . With a bit of calculation, one can note that the coalgebra  $L(f)(i_X)$  is defined by the composite

$$C \xrightarrow{\text{inr}\gamma} FX + Y + FC \xrightarrow{Y+[F\text{inl}, F\text{inr}]} Y + F(X + C) \xrightarrow{Y+F[f^+, C]} Y + FC$$

and the identity on  $C$  is clearly a coalgebra morphism between the two coalgebras (which is accepted because it's monic). Also, the  $i_X$ -th component of the natural transformation  $f^L$  turns out to be precisely  $f^+$ , therefore, by putting  $k = 1_C$ , the equality  $k.(f^L)_{i_X} = f^+$  follows immediately.

Finally, for  $Y, Z, g$  and  $y : A \longrightarrow Y + FA$  as in the theorem, we need to show that  $L(g).L(y) = L(L(g)y).\mathcal{I}(g_y^L)$  and  $g^L \diamond y^L = (L(g)y)^L \diamond 1$ . Let  $\beta : B \longrightarrow A + FB$  be a coalgebra in  $\mathcal{I}U_y = \mathcal{I}A$ . Then,

$$\begin{aligned} L(g)L(y)\beta &: (B_S + A)_S + C \longrightarrow Z + F((B_S + A)_S + C) \\ L(L(G)y)\mathcal{I}(g_y^L)\beta &: B_S + A_S + C \longrightarrow Z + F(B_S + A + S + C) \end{aligned}$$

and, with a bit of calculation, one can show that the two coalgebras are equal. Crucially, for this, one needs showing that  $(B_S + A)_S = B_S + A_S$ , but this follows by some simple pullback pasting. And last, the two components of the natural transformations both turn out to be the same map  $B_S + (A_S + g^+ \alpha_V)\beta_V : B \longrightarrow B_S + A_S + C$ , thus allowing us to state the following result:

**Proposition 4.7** *The functor sending  $X$  to  $\text{colim}_{\mathcal{I}_G} X$  defines a monad.*

### 4.3 Coalgebraicity Results

We have constructed a general theorem for defining monads as pointwise colimits. Since our overall interest lies in coalgebraic monads, the question arises as to whether it is possible to extend Theorem (3.10) to produce coalgebraic monads. Here we provide preliminary results showing that the term graph and rational monads are coalgebraic.

**Lemma 4.8** *Let  $F : \mathcal{C} \longrightarrow \mathcal{C}$  be a finitary functor,  $\mathcal{I}X$  a filtered category and  $H : \mathcal{I}X \longrightarrow \mathcal{I}X$  be a functor. If there is a natural transformation  $\alpha : FU_X \Rightarrow U_X H$ , then  $\text{colim} U_X$  is an  $F$ -algebra. Similarly if there is a natural transformation  $\alpha : U_X \Rightarrow FU_X H$ , then  $\text{colim} U_X$  is an  $F$ -coalgebra.*

**Proof.** By lemma 3.4, we have the map  $(H, \alpha) \in \mathbf{Lax}_{\mathcal{C}}(FU_X, U_X)$  and hence  $\text{colim} \alpha : \text{colim} FU_X = F \text{colim} U_X \longrightarrow \text{colim} U_X$  where the first equality is the finitariness of  $F$ . The second half of the theorem is proved analogously.  $\square$

We now prove that the rational and term graph monads are pointwise fixed points. To do this, we use the following property of coalgebras

**Lemma 4.9** *Let  $F : \mathcal{C} \longrightarrow \mathcal{C}$  be a functor and  $X$  and object of  $\mathcal{C}$ . Then, the mapping sending an  $X + F$ -coalgebra  $h : S \longrightarrow X + FS$  to the  $X + F$ -coalgebra  $X + Fh : X + FS \longrightarrow X + F(X + FS)$  defines a functor*

$$F_X : (X + F)\text{-coalg} \longrightarrow (X + F)\text{-coalg}.$$

Furthermore,  $(X + F \circ -).U_X = U_X F_X$ , there is a natural transformation  $\alpha : U_X \Rightarrow (X + F \circ -).U_X$  and another natural transformation  $\beta : 1 \Rightarrow F_X$  such that  $\alpha \diamond 1 = U\beta$ .

**Proof.** For the transformation  $\alpha$ , for an  $X + F$ -coalgebra  $\langle A, h \rangle$ , set  $\alpha_h$  to be the map  $h$  which is clearly a map of the right form. Naturality of  $\alpha$  is precisely the condition on a map  $k : S \longrightarrow S'$  such that  $k : (S, h) \longrightarrow (S', h')$  is a coalgebra homomorphism. Similarly, set  $\beta_h = h$ .  $\square$

**Lemma 4.10** *If  $F$  preserves finitely presentable objects, the rational and term graph monads are  $F$ -coalgebraic monads.*

**Proof.** We consider the rational monad, with the case of the term graph monad being analogous. The functor  $F_X$  preserves finite coalgebras and hence restricts to a functor  $\mathcal{I}_R X \longrightarrow \mathcal{I}_R X$  which we also denote  $F_X$ . Similarly,  $\alpha$  and  $\beta$  restrict to natural transformations  $\alpha_R$  and  $\beta_R$ . Since the functor  $X + F \circ -$  is finitary, by lemma 4.8,  $RX$  is both an  $X + F$ -coalgebra, via  $\text{colim}\alpha_R : RX \longrightarrow X + FRX$ , and an  $X + F$ -algebra, with structure map  $\text{colim}1 : X + FRX \longrightarrow RX$ .

$$\begin{array}{ccc} \mathcal{I}_R X & \xrightarrow{F_X} & \mathcal{I}_R X \\ U \downarrow & = & \downarrow U \\ \mathcal{C} & \xrightarrow{X + F \circ -} & \mathcal{C} \end{array} \quad \begin{array}{ccc} \mathcal{I}_R X & \xrightarrow{1} & \mathcal{I}_R X \\ U \downarrow & \Rightarrow_{\alpha_R} & \downarrow U \\ \mathcal{C} & \xleftarrow{X + F \circ -} & \mathcal{C} \end{array}$$

We use lemma 3.4 to show that these maps are mutually inverse. In one direction,  $(\text{colim}\alpha).(\text{colim}1) = \text{colim}(\alpha \diamond 1) = \text{colim}(U\beta) = \text{colim}1 = 1$ , where the second equality is from lemma 4.9 and all the others are from lemma 3.4. In the reverse direction, calculating the pasting gives  $1 \diamond \alpha_R = \alpha_R F_X = (X + F)U\beta$ , where the last equality holds since both natural transformations have, as component for a coalgebra  $h$ , the arrow  $X + Fh$ . Thus

$$\begin{aligned} (\text{colim}1).(\text{colim}\alpha) &= \text{colim}1 \diamond \alpha = \text{colim}\alpha F_X = \text{colim}(X + F)U\beta \\ &= X + F(\text{colim}U\beta) = X + F(1) = 1 \end{aligned}$$

where the fourth equality holds because  $X + F$  is finitary.

The family of maps making  $RX$  an  $X + F$ -coalgebra can be seen to be natural by noting that  $Rf$  arises via lemma 3.4 as  $\text{colim}f^R$  and  $X + Ff$  arises as  $\text{colim}X + F(f^R)$  where  $f^R = i_Y^R \diamond 1$ . Thus, commutation of the naturality

square amounts to proving that  $\alpha \diamond f^R = X + F(f^R) \diamond \alpha$ . At an  $X + F$ -coalgebra  $(S, h)$ ,  $(\alpha \diamond f^R)_h = L_{i_Y} \cdot \text{inl}$  while  $(X + F(f^R) \diamond \alpha)_h = X + F(\text{inl}) \cdot h$ . The definition of  $L_{i_Y}$  shows that these are equal. The naturality of the maps making  $RX$  an  $X + F$ -algebra follows from the naturality of the maps making  $RX$  an  $X + F$ -coalgebra and the fact that these maps are mutually inverse.

The first component of the algebra structure is the unit of  $R$  since the map  $\beta_{i_X} : i_X \longrightarrow F_X(i_X)$  induces the second of the following equalities  $\text{colim1} \cdot \text{inl} = \overline{F_X(i_x)} = \overline{i_X} = \eta$ . Finally, equation (1) is another diagram chase.  $\square$

## 5 Recursion Over Coalgebraic Monads

We have seen how rational terms arise as solutions of equations represented categorically as maps  $\phi : E \longrightarrow T_\Sigma(X + E)$  for fixed sets  $X$  and  $E$ . A solution for  $\phi$  is a map  $\phi^\dagger : E \longrightarrow T^\nu(X)$  such that the appropriate diagram commutes. In this section, we show that this ability to solve equations is possessed by all coalgebraic monads, and not just the initial one.

**Definition 5.1** Let  $H$  be an  $F$ -coalgebraic monad. An  $H$ -rational equation is a map  $\phi : E \xrightarrow{\phi} X + FH(X + E)$ .

A solution for a  $H$ -rational equation  $\phi$  is a map  $\phi^\dagger : E \longrightarrow T^\nu(X)$  for which

$$\begin{array}{ccc} E & \xrightarrow{\phi} & X + FH(X + E) \\ & \searrow \phi^\dagger & \swarrow \psi \\ & T^\nu(X) & \end{array}$$

where  $\psi \cdot \text{inl} = \eta$  and  $\psi \cdot \text{inr}$  is obtained as the composite

$$FH(X + E) \xrightarrow{F!} FT^\nu(X + E) \xrightarrow{FT^\nu[\eta, \phi]} FT^{\nu^2}(X) \xrightarrow{F\mu} FT^\nu \longrightarrow T^\nu$$

Notice that each  $H$ -rational equation  $\phi : E \xrightarrow{\phi} X + FH(X + E)$  determines a  $X + F$ -coalgebra structure for  $H(X + E)$

$$H(X + E) \xrightarrow{[\eta, \alpha]^{-1}} X + E + FH(H + E) \xrightarrow{[\text{inl}, \phi, \text{inr}]} X + FH(H + E)$$

Moreover, there is a bijective correspondence between solutions  $E \longrightarrow T^\nu(X)$  and  $X + F$ -coalgebra morphisms  $H(X + E) \longrightarrow T^\nu(X)$ . In one direction, we simply restrict a coalgebra morphism with the canonical inclusion  $E \longrightarrow H(X + E)$ , while given a map  $\psi : E \longrightarrow T^\nu(X)$  we can construct a map  $H(X + E) \xrightarrow{H[\eta, \psi]} HT^\nu(X) \longrightarrow T^\nu(X)$ . It's then simple diagram chasing to show that, under these constructions, being a solution corresponds to being a coalgebra map. We believe that the formulation of a solution via coalgebra maps is both cleaner and conceptually simpler than the usual formulation. Either way, since  $T^\nu(X)$  is the final  $X + F$ -coalgebra we have proved:

**Lemma 5.2** Every  $H$ -rational equation map  $E \xrightarrow{\phi} X + FH(X + E)$  has a unique solution  $\phi^\dagger : H(X + E) \longrightarrow T^\nu(X)$

We can go further and build coalgebraic monad morphisms into the picture

**Lemma 5.3** *Let  $\psi : H \longrightarrow H'$  be a morphism of coalgebraic monads. Then there is a map  $\psi^*$  sending  $H$ -rational equations to  $H'$ -rational equations such that for all  $H$ -rational equations  $\phi$ ,  $\phi^\dagger = (\psi^*(\phi))^\dagger.\psi$*

**Proof.** Given an  $H$ -rational equation  $E \xrightarrow{\phi} X + FH(X + E)$ , set  $\psi^*(\phi)$  to be the  $H'$ -rational equation  $E \xrightarrow{\phi} X + FH(X + E) \xrightarrow{X + F\psi} X + FH'(X + E)$ . Next, note that  $\psi$  defines a coalgebra morphism between the coalgebras  $[\text{inl}, \phi, \text{inr}]$  and  $[\text{inl}, X + F\psi.\phi, \text{inr}]$  which generate the solutions of  $\phi$  and  $\psi^*(\phi)$ . By the finality of  $T^\nu(X)$ , we therefore have  $\phi^\dagger = (\psi^*(\phi))^\dagger.\psi$   $\square$

The importance of this result is that it gives a simple, yet precise, proof of part of the implementation of functional languages using term graphs. For example, suppose we write  $c = \mathbf{A}(x, c, c)$  in a functional language. This is an equation written in the free monad which, as with all coalgebraic monads, has as solution an infinite term which could be taken to be its semantics. However, a compiler would turn this equation into the equation over term graphs mapping  $x$  to the term graph for  $\mathbf{A}(x, c, c)$  which shares the two copies of  $c$ . Thus we must ask how the solution to this term graph equation compares with the solution to the original equation. Lemma 5.3 assures us that the compiler will behave correctly in that solving the equation directly or solving the associated term graph equation produces the same result.

## References

- [1] P. Aczel, J. Adámek, and J. Velebil. A coalgebraic view of infinite trees and iteration. In Corradini et al., editor, *Proceedings of CMCS'01*, volume 44(1) of *ENTCS*. Elsevier, Amsterdam, 2001.
- [2] J. Adamek, S. Milius, and J. Velebil. Free iterative theories: a coalgebraic view. *submitted*, 2001.
- [3] J. Adamek and J. Rosický. *Locally Presentable and Accessible Categories*. Number 189 in London Mathematical Society Lecture Note Series. Cambridge University Press, 1994.
- [4] H. P. Barendregt, M. C. J. D. van Ekele, J. R. W. Glauert, J. R. Kennaway, M. J. Plasmeijer, and M. R. Sleep. Term graph rewriting. In *Proc. PARLE 87*, number 259 in LNCS, pages 141–158. Springer Verlag, 1987.
- [5] Aurelio Carboni, Steve Lack, and R.F.C. Walters. Introduction to extensive and distributive categories. *Journal of Pure and Applied Algebra*, 84:145–158, 1993.
- [6] A. Corradini and F. Gadducci. A 2-categorical presentation of term graph rewriting. In *Proceedings CTCS'97*, volume 1290 of LNCS. Springer, 1997.

- [7] Bruno Courcelle. Fundamental properties of infinite trees. *Theoretical Computer Science*, 25(2):95–169, 1983.
- [8] E. J. Dubuc and G. M. Kelly. A presentation of topoi as algebraic relative to categories or graphs. *Journal for Algebra*, 81:420–433, 1983.
- [9] C.C. Elgot, S.L. Bloom, and R. Tindell. On the algebraic structure of rooted trees. *Journal of Computing System Sciences*, 16:361–399, 1978.
- [10] M. Fiore, G. Plotkin, and D. Turi. Abstract syntax and variable binding. In *Proc. LICS'99*, 1999.
- [11] Neil Ghani, Cristoph Lüth, Federico De Marchi, and John Power. Algebras, coalgebras, monads and comonads. In Corradini et al., editor, *Proceedings of CMCS'01*, volume 44(1) of *ENTCS*. Elsevier, Amsterdam, 2001.
- [12] Masahito Hasegawa. *Models of Sharing Graphs: A Categorical Semantics of let and letrec*. PhD thesis, University of Edinburgh, 1999.
- [13] G. M. Kelly and A. J. Power. Adjunctions whose counits are coequalizers, and presentations of finitary monads. *Journal for Pure and Applied Algebra*, 89:163–179, 1993.
- [14] C. Lüth and N. Ghani. Monads and modular term rewriting. In *Category Theory in Computer Science CTCS'97*, number 1290 in LNCS, pages 69–86. Springer, 1997.
- [15] Lawrence Moss. Parametric corecursion. preprint, available at <http://math.indiana.edu/home/moss/parametric.ps>.