Under consideration for publication in Math. Struct. in Comp. Science

Dualizing Initial Algebras

NEIL GHANI[1], CHRISTOPH LÜTH[2],

FEDERICO DE MARCHI[1†], JOHN POWER[3]

[1] *Department of Mathematics and Computer Science, University of Leicester*

[2] *FB 3 — Mathematik und Informatik, Universität Bremen*

[3] *Laboratory for Foundations of Computer Science, University of Edinburgh*

Received 8 January 2003

Whilst the relationship between initial algebras and monads is well-understood, the relationship between final coalgebras and comonads is less well explored. This paper shows that the problem is more subtle than might appear at first glance: final coalgebras can form monads just as easily as comonads and dually, initial algebras form both monads and comonads.

In developing these theories we strive to provide them with an associated notion of syntax. In the case of initial algebras and monads this corresponds to the standard notion of algebraic theories consisting of signatures and equations: models of such algebraic theories are precisely the algebras of the representing monad. We attempt to emulate this result for the coalgebraic case by defining a notion of cosignature and coequation and then proving that the models of such coalgebraic presentations are precisely the coalgebras of the representing comonad.

1. Introduction

While the theory of coalgebras for an endofunctor is well-developed, less attention has been given to *comonads*. We feel this is a pity since the corresponding theory of monads on **Set** explains the key concepts of universal algebra such as *signature, variables, terms, substitution, equations* etc. Moreover, applications to base categories other than **Set** has proven fruitful in many situations, e.g. the study of S-sorted algebraic theories as monads over $\mathbf{Set}^S$, the study of inequational theories as monads over **Pos** (Robinson, 1994), the study of categories with structure using monads over **Graph** or **Cat** (Dubuc and Kelly, 1983), the study of rewriting using monads over **Pre** or **Cat** (Lüth, 1996; Lüth and Ghani, 1997) and the study of higher order abstract syntax using monads

† Research supported by EPSRC grant GRM96230/01: *Categorical Rewriting: Monads and Modularity*

over $\mathbf{Set}^{\mathbb{F}}$ (Fiore et al., 1999) (where $\mathbb{F}$ is the skeleton of the category of finite sets). We aim for a similar framework based on the theory of coalgebras and comonads which explains the relationship between the coalgebraic duals of the syntactic concepts and their semantic interpretation. Such a project is clearly beyond the scope of a single paper but, nevertheless, the constructions and applications contained within this paper lay the groundwork, and suggest this is likely to be a fruitful area of further research.

In initial algebra semantics, one traditionally begins with a signature Σ and then, for each set X of variables, defines the term algebra $T_\Sigma(X)$. Next, one defines substitution and shows this operation to be associative with the variables acting as left and right units. Finally, given equations E, one also defines the quotient algebra $T_{\langle\Sigma,E\rangle}(X)$. Generalising this treatment of universal algebra to cover not just sets with extra structure but also algebraic structure over other mathematical objects can be achieved by a categorical reformulation of these ideas. In the categorical framework (Kelly and Power, 1993), one first constructs from a signature Σ the free endofunctor F_Σ which can be thought of as mapping an object X of variables to the terms of depth 1 with variables from X. The term algebra T_Σ can then be characterised as the the free monad over F_Σ. Being a monad gives T_Σ a well behaved notion of substitution, while being free captures the inductive nature of the term algebra. By interpreting the equations of an algebraic theory as a pair of monad morphisms between two free monads, the quotient algebra induced by the equations is modelled as the coequaliser of the monad morphisms. Crucially, the models of an algebraic theory are isomorphic to the Eilenberg-Moore category of the monad representing the algebraic theory.

Our aim is to dualise this categorical treatment of universal algebra and in particular derive coalgebraic duals of the key concepts just described. However, there is more than one concept which can be dualised. As mentioned above, a signature Σ gives rise to a free endofunctor F_Σ, and the initial algebra of terms gives the free monad T_Σ. One dualisation is therefore to consider not initial algebras, but rather *final coalgebras*, while the other dualisation considers not the free functor and monad over a signature, but rather the *cofree functor and comonad* over the signature. Dualising the initial $X + F_\Sigma$-algebra of terms built over variables X to the final $X + F_\Sigma$-coalgebra gives not just finite terms, but the finite and infinite terms built over X. On the other hand, the final $X \times G_\Sigma$-coalgebra (or initial $X \times G_\Sigma$-algebra), where G_Σ is the *cofree* endofunctor generated from Σ, defines a comonad (and no terms, but something entirely different, as we will see below). In other words, given a signature Σ, we can consider its free endofunctor F_Σ, its cofree endofunctor G_Σ, and the following four mappings:

— the mapping sending X to the underlying object of the initial $X + F_\Sigma$-algebra;
— the mapping sending X to the underlying object of the final $X + F_\Sigma$-coalgebra;
— the mapping sending X to the underlying object of the initial $X \times G_\Sigma$-algebra;
— the mapping sending X to the underlying object of the final $X \times G_\Sigma$-coalgebra.

Each of these four mappings is an instance of the scheme in Table 1. That is, there is not one dualisation at play, but two orthogonal dualisations giving four concepts worthy of our attention. The first of these mappings corresponds to the traditional initial algebra

semantics described above while the essence of this paper is the consideration of the two final coalgebra constructions. For both constructions we prove that they indeed form monads and comonads respectively and seek to understand their associated notion of syntax. We show that the first dualisation corresponds to having a syntax containing not just finite terms, but also infinite terms while the second dualisation corresponds to a notion of syntax based upon predicate transformers as used in programming language semantics. In each instance we also sketch applications to more mainstream branches of computer science. The upper right corner of Table 1 (initial algebras as a comonad) remains unchartered for the time being; we leave its exploration to another paper.

	Monads	Comonads
Initial Algebras	$\mu Y.\, X + FY$	$\mu Y.\, X \times FY$
Final Coalgebras	$\nu Y.\, X + FY$	$\nu Y.\, X \times FY$

Table 1. *Algebras and Coalgebras for an endofunctor*

The rest of this paper is structured as follows: Section 2 contains preliminary definitions. In Section 3, we review the presentation of finitary monads which we seek to dualise in the rest of the paper (the upper left entry in Table 1). Sections 4 and 5 explore the different possibilities of dualisation, with Section 4 exploring the lower left of Table 1 and Section 5 containing our proposed comonadic syntax with associated representation and correctness results (the right side of Table 1). We conclude in Section 6 with remarks on further work and comments on the relationship between our work and the literature.

2. Preliminary Definitions and Notation

We assume the reader is familiar with standard category theory as can be found in (Mac Lane, 1971). In order to abstract away from the category **Set** we have had to employ certain constructions which some readers may not be completely familiar with. We give a short definition of these here; more details can be found in (Mac Lane, 1971; Adamek and Rosický, 1994).

Locally Presentable and Accessible Categories: Let κ be a regular cardinal. A diagram $\mathcal{D}$ is *κ-filtered* iff every subcategory with less than κ objects and morphisms has a compatible cocone in $\mathcal{D}$. An object X of a category $\mathcal{A}$ is said to be *κ-presentable* iff the hom-functor $\mathcal{A}(X, _)$ preserves κ-filtered colimits. A category is *locally κ-presentable* (abbreviated as lκp) if it is cocomplete and has a set N_κ of κ-presentable objects such that every object is a κ-filtered colimit of objects from N_κ. A category is κ-accessible providing it is lκp except that it only has κ-filtered colimits rather than all colimits. A functor is κ-accessible iff it preserves κ-filtered colimits; we also say it has *rank* κ. The discrete category on N_κ is denoted $\mathcal{N}_\kappa$. The full subcategory of κ-presentable objects is

denoted $\mathcal{A}_\kappa$. The inclusion functors are denoted $J_\kappa : \mathcal{N}_\kappa \to \mathcal{A}_\kappa$ and $I_\kappa : \mathcal{A}_\kappa \to \mathcal{A}$, and the category of κ-accessible functors from $\mathcal{A}$ to $\mathcal{B}$ by $[\mathcal{A}, \mathcal{B}]_\kappa$.

Finite presentability is the categorical notion for finiteness. For example, for $\mathcal{A} = \mathbf{Set}$, the finitely presentable sets are precisely finite sets, the set $N_{\aleph_0}$ can be taken to be the natural numbers which we denote $\mathbb{N}$, and every set is the filtered colimit of the diagram of all its finite subsets ordered by inclusion.

Kan Extensions: Given a functor $I : \mathcal{A} \to \mathcal{B}$ and a category $\mathcal{C}$, precomposition with I defines a functor $_ \circ I : [\mathcal{B}, \mathcal{C}] \to [\mathcal{A}, \mathcal{C}]$. The problem of left and right Kan extensions is the problem of finding left and right adjoints to $_ \circ I$. More concretely, given a functor $F : \mathcal{A} \to \mathcal{C}$, the left and right Kan extensions satisfy the natural isomorphisms

$$[\mathcal{B}, \mathcal{C}](\mathrm{Lan}_I F, H) \cong [\mathcal{A}, \mathcal{C}](F, H \circ I) \qquad [\mathcal{B}, \mathcal{C}](H, \mathrm{Ran}_I F) \cong [\mathcal{A}, \mathcal{C}](H \circ I, F).$$

Thus, one can view the left and right Kan extension of a functor $F : \mathcal{A} \to \mathcal{C}$ along $I : \mathcal{A} \to \mathcal{B}$ as the canonical extensions of the domain of F to $\mathcal{B}$. Kan extensions can be given pointwise using colimits and limits, or more elegantly using ends and coends (see (Mac Lane, 1971, Chapter X) for details).

3. Initial Algebras and Monads

We begin by firstly recalling the well known equivalence between (finitary) monads on the category **Set** and universal algebra and secondly by indicating the generalisation of this equivalence to locally presentable categories (Kelly and Power, 1993; Robinson, 1994). We illustrate these ideas with one example using **Set** as the base category and another using **Cat** as the base category — examples over other base categories can be found in (Robinson, 1994; Dubuc and Kelly, 1983).

A signature declares the operations from which the terms are constructed. Usually, a signature is given as a set of operations and a function assigning each operation an arity, but we can equivalently consider it as a function $\Sigma : \mathbb{N} \to \mathbf{Set}$, assigning to each arity the set of operations of that arity. In order to abstract away from the category of **Set**, we need a notion of arity appropriate for different categories. The key observation is that the usual arities in **Set**, i.e. the natural numbers $\mathbb{N}$, represent the finitely presentable objects in **Set**. Hence a natural notion for arities in a lκp-category is the set N_κ. Hence, a signature is a function $\Sigma : N_\kappa \to |\mathcal{A}|$, or equivalently a functor $\Sigma : \mathcal{N}_\kappa \to \mathcal{A}$:

Definition 3.1 (Signature). Let $\mathcal{A}$ be a lκp category. A *signature* is a functor $\Sigma : \mathcal{N}_\kappa \to \mathcal{A}$, and the object of *e-ary operations* of Σ is $\Sigma_e \stackrel{def}{=} \Sigma(e)$.

We now present two examples of such signatures, which we will later augment by the appropriate equations.

Example 3.2 (Monoids). The signature $\Sigma_M : \mathbb{N} \to \mathbf{Set}$ for the theory of monoids is defined as $\Sigma_M(0) = \{e\}, \Sigma_M(2) = \{m\}, \Sigma_M(n) = \emptyset$ for all other $n \in \mathbb{N}$. Thus Σ_M declares one operation e of arity 0 (a constant) and one binary operation m.

Example 3.3 (Categories with $\top$). Categories with a terminal object can be seen as algebraic over **Cat**. The signature $\Sigma_\top$ must declare the terminal object and for each object X, the unique map $!_X : X \to \top$. Since the terminal object does not depend upon any data, its arity is the empty category $\mathbf{0}$. Since the map $!_X$ depends upon exactly one object, its arity is the one object category $\mathbf{1}$. Thus $\Sigma_\top(\mathbf{0}) = \mathbf{1}$, $\Sigma_\top(\mathbf{1}) = (\circ \to \circ)$ (i.e. the category with two objects and one arrow) while $\Sigma_\top(c) = \mathbf{0}$ for all other finitely presentable categories c. Notice that the signature only declares operations; the equations will make the map unique, and force its domain to be X and its codomain to be $\top$.

We now turn to the construction of an endofunctor over a signature Σ. Working over **Set**, e.g. see (Turi, 1996), one defines a functor

$$F_\Sigma(X) = \coprod_{f \in \Sigma(n)} X^n$$

which calculates the terms of depth 1 whose variables come from the set X. For example, given the signature of example 3.2, the associated endofunctor is $F_M(X) = 1 + X^2$. Unfortunately, it is unclear how to generalise this construction of an endofunctor from a signature to categories other than **Set** and hence we take an alternative approach. Recall that by definition, the left Kan extension is the left adjoint to the restriction $\mathrm{Lan}_{J_\kappa} _ \dashv _ \circ J_\kappa : [\mathcal{N}_\kappa, \mathcal{A}] \to [\mathcal{A}_\kappa, \mathcal{A}]$ and so we define

$$F_\Sigma(X) = (\mathrm{Lan}_{J_\kappa} \Sigma) X. \tag{1}$$

The standard formula for left Kan extensions shows that, in the case $\mathcal{A} = \mathbf{Set}$, both definitions of F_Σ agree. For instance, in example 3.2, and using the classic coend formula:

$$\begin{aligned}(\mathrm{Lan}_{J_{\aleph 0}} \Sigma_M) X &= \coprod_{n \in \mathbb{N}} \mathbf{Set}_{\aleph 0}(n, X) \times \Sigma_M(n) \\ &= \mathbf{Set}(0, X) \times 1 + \mathbf{Set}(2, X) \times 1 \\ &= 1 + X^2\end{aligned}$$

Notice that the domain of F_Σ is $\mathcal{A}_\kappa$ and not $\mathcal{A}$, and hence F_Σ is not an endofunctor. However, an endofunctor $F : \mathcal{A} \to \mathcal{A}$ is κ-accessible iff it is (isomorphic to) the left Kan extension of its restriction to $\mathcal{A}_\kappa$. Thus there is an equivalence (2) between κ-accessible endofunctors on $\mathcal{A}$ and functors $\mathcal{A}_\kappa \to \mathcal{A}$. Consequently, we can regard F_Σ as a κ-accessible endofunctor.

$$[\mathcal{A}, \mathcal{A}]_\kappa \underset{\mathrm{Lan}_{I_\kappa} _}{\overset{_ \circ I_\kappa}{\underset{\top}{\rightleftarrows}}} [\mathcal{A}_\kappa, \mathcal{A}] \tag{2}$$

To recap, we have presented an abstract definition of signature Σ over an lκp-category and a construction of the associated endofunctor F_Σ over the signature. In the case of **Set**, these definitions and constructions agree with the usual definitions. The associated term algebra $T_\Sigma : \mathcal{A} \to \mathcal{A}$ is then the free monad over the endofunctor F_Σ, which can be constructed in a number of ways:

Lemma 3.4. Let F be a κ-accessible endofunctor over a lκp-category $\mathcal{A}$. Then the free monad T over F satisfies the following:

1 For every X in $\mathcal{A}$, TX is the carrier of the initial $X + F$-algebra.
2 The forgetful functor $U : F\text{–}\mathbf{Alg} \to \mathcal{A}$ from the category of F-algebras to $\mathcal{A}$ has a left adjoint L, and $T \cong UL$
3 T is the initial algebra of the functor $1 + F \circ _ : [\mathcal{A}, \mathcal{A}] \to [\mathcal{A}, \mathcal{A}]$
4 Regarding F as an endofunctor with domain $\mathcal{A}_\kappa$, T is the colimit of the sequence S_i

$$S_0 = I_\kappa \quad S_{n+1} = I_\kappa + F \diamond S_n \quad S_\lambda = \operatorname*{colim}_{i<\lambda} S_i \tag{3}$$

and the composition of $F, E : \mathcal{A}_\kappa \to \mathcal{A}$ is given as $F \diamond E \stackrel{def}{=} (Lan_{I_\kappa} F \circ E)$.

Proof. Most of the proofs are standard and can be found in (Kelly and Power, 1993; Kelly, 1980). However, note that in the last of the claims, to calculate the free monad we start forming the sequence of endofunctors S_i and we do not need to go further than the κ-filtered colimit $colim_{i<\kappa} S_i$ because F is κ-filtered. Hence T is κ-accessible and thus shares the same rank as F. □

When we come to the dual of this construction in Section 5, a construction like the above will not be possible since one would be interested in the limit of a co-chain, and there is no reason for an accessible endofunctor to preserve such limits. The effect is that the rank of the cofree comonad over an accessible endofunctor may increase. This change of rank underlies the technical difficulties which will arise in Section 5.

We obtain the two adjunctions in (4) which compose to give an adjunction $F \dashv U$ between signatures and monads, where $\mathbf{Mon}_\kappa(\mathcal{A})$ is the category of κ-accessible monads over $\mathcal{A}$:

$$[\mathcal{N}_\kappa, \mathcal{A}] \underset{_ \circ J_\kappa}{\overset{\mathrm{Lan}_{J_\kappa} _}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} [\mathcal{A}_\kappa, \mathcal{A}] \underset{V}{\overset{H}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathbf{Mon}_\kappa(\mathcal{A}) \tag{4}$$

The sequence S_i in (3) in Lemma 3.4 is called the *free algebra sequence* (Kelly, 1980) and can be seen as a uniform calculation of the initial $X + F$ algebra. As an example of this construction, consider the signature Σ_M. As we have seen $\mathrm{Lan}_{J_\kappa} \Sigma_M(X) = 1 + X^2$. The free algebra sequence then specialises to $S_0 X = X$ and

$$S_{n+1}(X) = X + \coprod_{e \in \mathbb{N}} \mathcal{A}(e, S_n(X)) \times \Sigma_M(e) = X + 1 + S_n(X)^2$$

from which we can see S_n as defining the terms of depth at most n, e.g. $S_0(X)$ contains the variables X, and $S_1(X)$ contains the variables X, the canonical element of 1 representing the unit of the monoid and a pair of elements of $S_0(X)$ which can be thought of as the multiplication of these elements.

Within this framework, to give equations is to give another signature E and two natural transformations $\sigma, \tau : E \to UFB$. One should regard E as giving the shape of the equations and σ and τ as the actual equations.

Example 3.5. Given the monoids example, there are three equations we wish to assert: left unit, right unit (both unary) and associativity (ternary). Hence we set $E(3) = \{a\}, E(1) = \{l, r\}$ and $E(n) = \emptyset$ for all other n. The natural transformations σ and τ define the left, and the right hand side of equations, respectively:

$$\begin{array}{lll} \sigma(l) = m(e, x) & \sigma(r) = m(x, e) & \sigma(a) = m(x, m(y, z)) \\ \tau(l) = x & \tau(r) = x & \tau(a) = m(m(x, y), z) \end{array}$$

For example 3.3, we need three equations: one equation to force the uniqueness of $!_X : X \to \top$, i.e. $!_Y \cdot f = !_X$ for all $f : X \to Y$, and two further equations to correctly state the source and target of $!_X$. See (Dubuc and Kelly, 1983) for details.

The advantage of this definition of equations is that it allows an elegant derivation of the representing monad for an algebraic theory: given such a theory $\sigma, \tau : E \to UFB$, under adjunction (4) we have two monad morphisms σ', τ' whose coequaliser is the representing monad T. This construction makes sense, because the category of models of the algebraic theory is isomorphic to the Eilenberg-Moore-category of the representing monad.

In (4), both adjunctions are monadic and their composition is of *descent type*, which means that each component of the counit $\varepsilon_B : FUB \to B$ is a coequaliser. This means that *every* κ-accessible monad T on $\mathcal{A}$ is a coequaliser in the category of κ-accessible monads over $\mathcal{A}$ of two free monads over two signatures $B, E : \mathcal{N} \to \mathcal{A}$. Equivalently, every such monad is represented by equations in this general sense.

4. Final Coalgebras and Monads

Given a signature Σ we have constructed the term algebra T_Σ as the free monad over Σ. Concretely, $T_\Sigma X$ is the carrier of the initial $X + F_\Sigma$-algebra which, when specialised to **Set**, corresponds to the set of finite terms whose variables are in X.

In this section we investigate the map T_Σ^∞ sending an object X to the carrier of the final $X + F_\Sigma$-coalgebra, comment on the notion of syntax inherent in this construction and apply our results to simplify some recent developments in the semantics of lazy datatypes in functional programming. A more general, and in our opinion more elegant, version of Lemma 4.2 appears in (Aczel et al., 2001) although the result first appears in (Moss, 1999). Nevertheless, we include our proof since it is significantly different in style to those mentioned above.

The first observation in comparing $\mu Y.X+FY$ with $\nu Y.X+FY$ was by Barr, showing that the final coalgebra of an endofunctor on **Set** can be regarded as a Cauchy completion of the initial algebra (Barr, 1993). Thus, if F arises from a signature and hence $\mu Y.X + FY$ is the usual term algebra, then $\nu Y.X + FY$ is easily seen to be the set of terms of finite and infinite depth. Barr's result was extended to lfp categories in (Adamek, 2000). The goal of this section is to ascertain the structure possessed by the map sending X to the carrier of the final $X + F_\Sigma$-coalgebra. Our answer is that this map also forms a monad. In proving this result we use the following result (Adamek, 2000):

Lemma 4.1. Assume that $\mathcal{A}$ is lfp, that the unique map $! : 0 \to 1$ is a strong monomorphism, that F preserves monos and ω^{op}-chains and that there is a map $p : 1 \to F0$. Then for each object $X \in \mathcal{A}$, $\mathcal{A}(X, \mu F)$ and $\mathcal{A}(X, \nu F)$ are metric spaces with the latter being the Cauchy completion of the former.

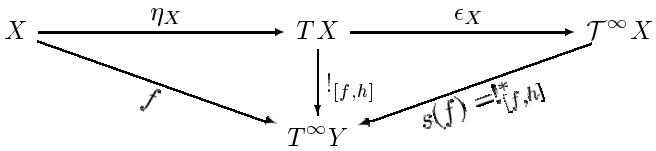
We use this result to prove our main lemma:

Lemma 4.2. Let F be a polynomial endofunctor and $\mathcal{A}$ satisfy the premises of Lemma 4.1. Then the map T^∞ assigning an object X to the carrier of the final $X+F$-coalgebra carries the structure of a monad.

Proof. We actually prove the lemma for the case $\mathcal{A} = \mathbf{Set}$ since the general case involves slightly clumsy reasoning with hom-sets but is entirely analogous. We prove that T^∞ is a monad by constructing a Kleisli structure for it, namely by defining the following operations: i) for each object X, an arrow $\eta^\infty_X : X \to T^\infty X$; ii) for each pair of objects X, Y a function $s_{X,Y} : \mathcal{A}(X, T^\infty Y) \to \mathcal{A}(T^\infty X, T^\infty Y)$ such that

$$s_{X,X}(\eta^\infty_X) = 1_{T^\infty X} \qquad s_{X,Z}(s_{Y,Z}g \circ f) = s_{Y,Z}g \circ s_{X,Y}f \qquad s_{X,Y}(f) \circ \eta^\infty_X = f$$

Firstly define $\eta^\infty_X = \epsilon_X \circ \eta_X$ where η is the unit of the free monad on F and ϵ_X is the unique comparison map between the initial $X+F$-algebra and the final $X+F$-coalgebra.[†] Next define $s_{X,Y}(f)$ as in (5)



(5)

where $h : FT^\infty Y \to T^\infty Y$ is the obvious map derived from the structure map of $T^\infty Y$. Then $[f, h] : X + FT^\infty Y \to T^\infty Y$ endows $T^\infty Y$ with an $X + F$-algebra structure and hence induces a unique map $!_{[f,h]} : TX \to T^\infty Y$ from the initial $X + F$-algebra. Diagram chasing shows that $!_{[f,h]}$ is uniformly continuous with respect to the relevant metrics and hence by Lemma 4.1 there is an extension $!^*_{[f,h]}$ which can be taken to be $s_{X,Y}(f)$. The first equation follows since by uniqueness $!_{[\eta^\infty_X, h]} = \epsilon_X$ and the extension of ϵ_X is the identity. The equation $s_{X,Z}(s_{Y,Z}g \circ f) = s_{Y,Z}g \circ s_{X,Y}f$ holds if their restrictions via ϵ_X are the same which in turn holds if they arise from the same algebra structure on $T^\infty Z$. This holds because the two algebra structures are both inherited from the map $s_{Y,Z}(g) \circ f : X \to T^\infty Z$. The last equation follows since $s_{X,Y}(f) \circ \eta^\infty_X = s_{X,Y}(f) \circ \epsilon_X \circ \eta_X = !_{[f,h]} \circ \eta_X = f$. □

Notice that Lemma 4.2 indicates that there is no need to develop a new syntax for the monad T^∞_Σ since the canonical syntax should be the finite and infinite depth terms of the associated monad T_Σ. Unlike the monad T, T^∞ has a larger rank than F:

[†] Actually applying Lemma 4.1 requires that $X + F0$ be non empty which is the case when $X \neq 0$. Thus the maps η^∞_0 and $s_{0,Y}$ and their properties must be established separately, but this all follows trivially from the initiality of 0.

Lemma 4.3. Let F be a polynomial endofunctor on **Set**. Then the monad T^∞ given by Lemma 4.2 has rank $\aleph_1$.

Proof. The lemma can be proven directly using the interchange law between limits and filtered colimits (Mac Lane, 1971, Theorem IX.2.1). □

Incorporating equations into this coalgebraic setting or, more precisely, defining the quotient of $T^\infty(X)$ by a system of equations is not straightforward. Indeed, the following example shows that equational theories which have a consistent initial semantics may have an inconsistent final semantics. Consider an algebraic theory consisting of a unary symbol A, a constant B and the equation $A(x) = x$. The initial algebra semantics of this algebraic theory maps a set X to the set $X + 1$ since, after quotienting by the equations, all that is left is the variables and the constant B. But, in the final coalgebra, there are infinite terms and hence we have $x = A(x) = A^2(x) = \ldots = A^\omega$. Hence we can prove that, for another variable y, $y = A^\omega$ and hence that $x = y$. Since these variables are arbitrary, this allows us to prove that any two terms are equal and hence that the associated equational theory is inconsistent. Controlling this phenomenon is an area of ongoing research.

We finish this section with an application of this result to reasoning in functional programming.

The Generic Approximation Lemma

The *generic approximation lemma* (Hutton and Gibbons, 2001) is a proof principle for reasoning about functions in a lazy functional programming language (such as Haskell). It pertains to lists and states that two potentially infinite lists `xs` and `ys` are equal iff $\forall n.$`approx` n `xs = approx` n `ys` where `approx` is defined by

```
approx (n+1) []     = []
approx (n+1) (x:xs) = x:(approx n xs)
```

Note the lack of a base case: `approx 0 x` is $\bot$ (i.e., the denotation of undefined) in the denotational model, but because of non-strictness, `approx` n `x` (with $n > 0$) is defined. This principle can be applied to other datatypes such as trees:

```
data Tree a = Leaf a | Node Tree a Tree
approx (n+1) (Leaf x) = Leaf x
approx (n+1) (Node l x r) = Node (approx n l) x (approx n r)
```

Two trees `t1` and `t2` are equal iff $\forall n.$ `approx` n `t1 = approx` n `t2`. Hutton and Gibbons (2001) prove the generic approximation lemma using the standard denotational semantics of functional programming languages, where types are interpreted as cpo's, programs as continuous functions and recursive datatypes as least fixed points of functors. That is, the correctness of the proof principle depends upon the semantic category chosen; we have already seen the implicit use of $\bot$ in the definition of `approx`.

We propose an alternative and, we believe, more natural derivation of the approximation lemma which is independent of the particular denotational model chosen. The definition of a polymorphic datatype is usually interpreted as the free monad T_Σ over its signature Σ. However, this does not capture laziness, since T_Σ consists of only finite terms. Instead, we model such a datatype by the monad T_Σ^∞. Since $T_\Sigma^\infty(X)$ is the final $X + F_\Sigma$-coalgebra and since F_Σ is a polynomial endofunctor, $T_\Sigma^\infty(X)$ can be calculated as the limit of the ω^{op}-chain $1 \leftarrow (X+F)1 \leftarrow (X+F)^2 1 \leftarrow \cdots$. The universal property of the limit states that two elements x and y of this limit will be equal iff for each n, $\pi_n(x) = \pi_n(y)$ where π_n is the n-th projection. But these projections are precisely the approximation function for the datatype. Notice how the categorical argument replaces the semantic dependency on $\perp$ with use of a co-chain beginning with 1. This establishes the correctness of the generic approximation lemma independently of any specific denotational model.

5. Final Coalgebras and Comonads

We now turn to another possible dualisation of Section 3 based upon the idea of mapping an object X to the carrier of the final $X \times F$-coalgebra for some κ-accessible functor F. The elegance of the monadic approach to algebraic theories suggests that we can apply these techniques to the development of coalgebraic syntax by dualising them appropriately.

5.1. *Cosignatures and their Coalgebras*

Recall that the heart of the categorical approach to universal algebra is adjunction (4). The dualisation outlined in this section can be summed up as replacing the left adjoint to U with a right adjoint and the replacement of monads with comonads. As a result, the definition of a cosignature turns out to be formally the same as that of a signature.

Definition 5.1 (Cosignature). Let $\mathcal{A}$ be a lκp-category with arities $\mathcal{N}_\kappa$. A κ-cosignature is a functor $B : \mathcal{N}_\kappa \to \mathcal{A}$.

Recall that in Section 3, we constructed a κ-accessible endofunctor from a signature by first taking a left Kan extension, and then used equivalence (2) to get a κ-accessible endofunctor. Therefore, in this section we take the right Kan extension of a cosignature B to obtain a functor $\mathrm{Ran}_{J_\kappa} B : \mathcal{A}_\kappa \to \mathcal{A}$, followed by the same equivalence to obtain a κ-accessible endofunctor on $\mathcal{A}$. However, recall that equivalence (2) is based on a left Kan extension; this mixture of left and right Kan extension makes the situation delicate.

The standard formula for the right Kan extension gives us

$$(\mathrm{Ran}_{J_\kappa} B)(X) = \prod_{c \in \mathcal{N}_\kappa} \mathcal{A}(X, c) \pitchfork Bc$$

where $U \pitchfork B$ is the U-fold product of B, or more formally the representing object for the functor $[U, \mathcal{A}(_, B)] : \mathcal{A} \to \mathbf{Set}$. When $\mathcal{A} = \mathbf{Set}$, then this operation is simply

exponentiation, i.e. $U \pitchfork B = [U, B]$. Thus, although signatures and cosignatures are formally the same, the endofunctors they generate are very different. For example, note that while the default value for signatures is 0, if there is a single arity c such that $B(c) = 0$, then $(\mathrm{Ran}_{J_\kappa} B)(X) = 0$. In fact, the default value for cosignatures is 1 since $U \pitchfork 1 = 1$ and hence if c is an arity such that $B(c) = 1$, then this arity will contribute nothing to the right Kan extension. Here are two example cosignatures which we will explore further below:

Example 5.2. Define the cosignature B_p by $B_p(2) = 2$ and $B_p(c) = 1$ for all other arities. Then $\mathrm{Ran}_{J_\kappa} B_p(X) = [\mathbf{Set}(X, 2), 2] = [[X, 2], 2]$.

Define the cosignature $B_{p_\omega}(2) = \omega$ and $B_{p_\omega}(c) = 1$ for all other arities. Then, by a similar calculation, $(\mathrm{Ran}_{J_\kappa} B_{p_\omega})(X) = [\mathbf{Set}(X, 2), \omega] = [[X, 2], \omega]$.

Recall that in order to turn a functor $\mathcal{A}_\kappa \to \mathcal{A}$ into a κ-accessible endofunctor, one takes its left Kan extension. Given a cosignature B, we can thus consider coalgebras of the endofunctor $\mathrm{Lan}_{I_\kappa} \mathrm{Ran}_{J_\kappa} B$. The following is an important result from (Johnstone et al., 1998):

Lemma 5.3. Let $F : \mathcal{A} \to \mathcal{A}$ be an accessible endofunctor on an accessible category. Then F–**coalg** is accessible.

Proof. The proof uses the fact that the category of all (small) accessible categories has all weighted limits (Makkai and Paré, 1989, Theorem 5.1.6). Then the category F–**coalg** can be constructed as such a limit, namely the *inserter* (Kelly, 1989) of the diagram

$$\mathcal{C} \underset{F}{\overset{1}{\rightrightarrows}} \mathcal{C}$$

□

Given that F–**coalg** is accessible, we would like to know its degree of accessibility but this depends upon a number of rather technical properties (Adamek and Rosický, 1994). What is true is that there is a cardinal κ such that F–**coalg** is κ-accessible and that an F-coalgebra is κ-presentable iff its underlying object is κ-presentable. Perhaps of more use is the result that if F and $\mathcal{A}$ have rank λ, then a coalgebra whose underlying object is λ-presentable is itself λ-presentable. This result seems to be known although we could not find a reference.

The next step is to compare $\mathrm{Lan}_{I_\kappa} \mathrm{Ran}_{J_\kappa} B$-coalgebras with a notion of model for the underlying cosignature. As remarked above, the interaction of left and right Kan extensions makes a general comparison impossible — our solution is to consider only those coalgebras which are small enough to negate the effect of the left Kan extension.

Lemma 5.4. Let B be a κ-cosignature. A $\mathrm{Lan}_{I_\kappa} \mathrm{Ran}_{J_\kappa} B$-coalgebra over a κ presentable object X consists of, for every arity c, a function $h_c : \mathcal{A}(X, c) \to \mathcal{A}(X, Bc)$.

Proof. Given any κ-accessible endofunctor F, and κ-presentable object X, $\mathrm{Lan}_{I_\kappa} F(X) = F(X)$. We then have the following chain of isomorphisms:

$$\begin{array}{rcl} \mathcal{A}(X, \mathrm{Lan}_{I_\kappa}(\mathrm{Ran}_{J_\kappa} B)(X)) & \cong & \mathcal{A}(X, (\mathrm{Ran}_{J_\kappa} B)(X)) \\ & \cong & \mathcal{A}(X, \prod_{c \in \mathcal{N}_\kappa} \mathcal{A}(X, c) \pitchfork Bc) \\ & \cong & \prod_{c \in \mathcal{N}_\kappa} \mathcal{A}(X, \mathcal{A}(X, c) \pitchfork Bc) \\ & \cong & \prod_{c \in \mathcal{N}_\kappa} [\mathcal{A}(X, c), \mathcal{A}(X, Bc)] \end{array}$$

□

In the theory of monads, a crucial result states that the models of an algebraic presentation are isomorphic to the category of algebras of the representing monad. In order to emulate this result we require a notion of model (or co-model if you like) for a κ-cosignature B, and Lemma 5.4 suggests this be $\mathrm{Lan}_{I_\kappa} \mathrm{Ran}_{J_\kappa} B$-coalgebra.

Example 5.5. A model of the cosignature B_p is given by a finite set X together with a function $\mathbf{Set}(X, 2) \to \mathbf{Set}(X, 2)$. If we regard a map $f : X \to 2$ as a subset of X, or a predicate over X, then such a model can be regarded as a map between predicates.

Similarly, a model for the cosignature B_{p_ω} consists of a finite set X and a function $\mathbf{Set}(X, 2) \to \mathbf{Set}(X, \omega)$. This is clearly related to the previous example if we think of it as mapping binary partitions of X to ω-ary partitions.

5.2. *From Coalgebras to Predicate Transformer Semantics*

Example 5.5 exhibited maps between predicates as coalgebras. A natural application of our research on coalgebras would then seem to be the semantics of programming languages based upon what are known as *predicate transformers*. However, to make this idea precise, as we shall see, a bit more structure is required.

The basic idea of predicate transformer semantics going back to (Dijkstra, 1976) is that the meaning of a program is a map sending a postcondition to its weakest precondition. One starts with a set of states (e.g. a mapping from variable locations to integers) and then considers pre- and post conditions as predicates over the set of states. Thus the semantics of a program is a mapping from predicates to predicates, hence the term predicate transformer. However, in order to account for recursion and iteration, predicate transformers must be continuous which in turn means that the set of predicates must form some order, e.g. an ω-cpo. (see e.g. (Winskel, 1993, Section 7.5) for details).

In order to cater for this order structure, we move from the category **Set** to the category ω**CPO** of ω-complete partial orders (ω-cpo's) and continuous maps. Unlike **Set**, the category ω**CPO** is not locally finitely presentable, but locally $\aleph_1$-presentable, as the operation of taking the least upper bound of an ω-chain is an infinitary operation (Adamek and Rosický, 1994). This is another example of the necessity not to restrict ourselves to **Set**, or even to locally finitely presentable categories.

We can now easily adapt Example 5.5 by changing the base category to ω**CPO**. Let the

$\aleph_1$-cosignature $B_{pred} : \mathcal{N} \to \omega\mathbf{CPO}$ be defined by $B_{pred}(\mathbf{2}) = \mathbf{2}$ and $B_{pred}(c) = \mathbf{1}$ (where $\mathbf{1}$ is the one-element ω-cpo) for all other arities. By Lemma 5.4, a coalgebra for this signature over a $\aleph_1$-presentable object consists of a countable $\omega\mathbf{CPO}$ X (thought of as a set of states) together with a map $\omega\mathbf{CPO}(X, \mathbf{2}) \to \omega\mathbf{CPO}(X, \mathbf{2})$. The continuity of the predicate transformers is therefore guaranteed while the representation of predicates as continuous maps $X \to \mathbf{2}$ agrees with the traditional approach where states are unordered or, in effect, discretely ordered.

This is of course only a sketch of the use of coalgebras in predicate transformer semantics, as any substantial treatment is beyond the scope of this paper.

5.3. *The Representing Comonad of a Cosignature*

Recall that in the algebraic case one starts with a signature, constructs an endofunctor, and then considers the free monad over the endofunctor. Having dualised the first part of this construction, we now dualise the second, i.e. consider the construction of the cofree comonad over an endofunctor. The following pair of results from (Johnstone et al., 1998) provides the framework for this discussion:

Lemma 5.6. The following conditions on a functor $F : \mathcal{A} \to \mathcal{A}$ are equivalent:

1 There is a cofree comonad on F.
2 The forgetful functor F–**coalg** $\to \mathcal{A}$ is comonadic.
3 The forgetful functor F–**coalg** $\to \mathcal{A}$ has a right adjoint
4 (If $\mathcal{A}$ has products) For every object X, the functor $X \times F$ has a final coalgebra.

Similar to the monadic case, not all endofunctors have a cofree comonad. For example, there is no final coalgebra for the powerset functor $P : \mathbf{Set} \to \mathbf{Set}$, as such a coalgebra would be an isomorphism $X \cong PX$ which cannot exist by cardinality arguments. The important result from (Makkai and Paré, 1989; Johnstone et al., 1998) is the following:

Lemma 5.7. If $\mathcal{A}$ is locally presentable and F is accessible, then F–**coalg** is locally presentable and there is a cofree comonad over F.

Proof. Recall from Lemma 5.3 that F–**coalg** is accessible. Cocompleteness of F–**coalg** follows since the forgetful functor creates colimits and $\mathcal{A}$ is cocomplete. Since the forgetful functor is a colimit-preserving functor between locally presentable categories, it follows from the Special Adjoint Functor Theorem (Mac Lane, 1971, Theorem V.8.2) that the forgetful functor has a right adjoint. By Lemma 5.6, there is a cofree comonad over F. □

Let $\mathbf{Com}_\kappa(\mathcal{A})$ be the category of κ-accessible comonads on $\mathcal{A}$, and further $\mathbf{ACom}(\mathcal{A})$ be the category of accessible comonads on $\mathcal{A}$. For every κ, $\mathbf{Com}_\kappa(\mathcal{A})$ is a full subcategory of $\mathbf{ACom}(\mathcal{A})$. Recall that $[\mathcal{A}, \mathcal{A}]_\kappa$ is the category of endofunctors with rank κ, and let $\mathbf{AEnd}(\mathcal{A})$ be the category of accessible endofunctors. Piecing together the above results we have the following:

Lemma 5.8. The forgetful functor $V : \mathbf{ACom}(\mathcal{A}) \to \mathbf{AEnd}(\mathcal{A})$ has a right adjoint R.

Proof. Given an accessible endofunctor F, let the right adjoint of the forgetful functor $U_F : F\text{–}\mathbf{coalg} \to \mathcal{A}$ be K_F. Then choosing $R(F)$ to be $U_F K_F$ we can deduce that $R(F)$ is accessible because U_F is accessible (as a left adjoint, it preserves all colimits) and K_F is accessible by (Adamek and Rosický, 1994, Proposition 1.66 on p. 52).

The action of the right adjoint on accessible endofunctors has already been given. Given a natural transformation $\alpha : F_1 \Rightarrow F_2$, this induces a functor $\alpha^* : F_1\text{–}\mathbf{coalg} \to F_2\text{–}\mathbf{coalg}$ which commutes with the respective forgetful functors. Using the dual of (Barr and Wells, 1985, Theorem 3 on p. 128) and the comonadicity of these categories of coalgebras, this in turn induces a natural transformation $R(\alpha) : R(F_1) \Rightarrow R(F_2)$. This defines the functor $R : \mathbf{AEnd}(\mathcal{A}) \to \mathbf{ACom}(\mathcal{A})$. The fact that it is right adjoint to V is easily verified. □

As opposed to the monadic case, the cofree comonad RM on an endofunctor $M : \mathcal{A} \to \mathcal{A}$ of rank λ need not have rank λ; as a simple counterexample, consider the endofunctor $M : \mathbf{Set} \to \mathbf{Set}$ defined as $MX \stackrel{def}{=} A \times X$ for a fixed set A, which clearly is finitary (has rank $\aleph_0$). We know that the value of RM for a set X is given by $RM(X) = \nu Y.\, X \times MY = \nu Y.\, X \times A \times Y$ which is the set of all infinite streams of elements of $X \times A$. Now consider a countably infinite set Y, then $RM(Y)$ contains a sequence with infinitely many different elements from Y. This sequence can not be given from any of the finite subsets $Y_0 \subseteq Y$, which shows that RM has a rank larger than $\aleph_0$. As we saw in Section 4, in general calculating coalgebras of accessible endofunctors increases their rank — see (Worrell, 1999) for an investigation of this phenomenon.

Using the equivalence between $[\mathcal{A}_\kappa, \mathcal{A}]$ and $[\mathcal{A}, \mathcal{A}]_\kappa$, we now have the following functors, where $_ \circ J_\kappa$ is the precomposition with J_κ and V is the forgetful functor for comonads of rank κ:

$$\begin{array}{ccccc} [\mathcal{N}_\kappa, \mathcal{A}] & \underset{_ \circ J_\kappa}{\overset{\mathrm{Ran}_{J_\kappa}}{\rightleftarrows}} & [\mathcal{A}_\kappa, \mathcal{A}] & \cong & [\mathcal{A}, \mathcal{A}]_\kappa \xleftarrow{\quad V \quad} \mathbf{Com}_\kappa(\mathcal{A}) \\ & & & & \downarrow \qquad\qquad \downarrow \\ & & & & \mathbf{AEnd}(\mathcal{A}) \underset{V}{\overset{R}{\rightleftarrows}} \mathbf{ACom}(\mathcal{A}) \end{array} \tag{8}$$

We further define the composite functors

$$\begin{array}{ll} V_\kappa : \mathbf{Com}_\kappa(\mathcal{A}) \to [\mathcal{N}_\kappa, \mathcal{A}] & V_\kappa \stackrel{def}{=} (_ \circ J_\kappa) \cdot V \\ R_\kappa : [\mathcal{N}_\kappa, \mathcal{A}] \to \mathbf{ACom}(\mathcal{A}) & R_\kappa \stackrel{def}{=} R \cdot \mathrm{Ran}_{J_\kappa} \end{array}$$

As we have seen, the rank of $R_\kappa B$ may be greater than the rank of B, and hence the codomain of R_κ is not necessarily the domain of V_κ, but rather $\mathbf{ACom}(\mathcal{A})$. So although R_κ and V_κ can not be adjoint, there is the following isomorphism:

Lemma 5.9. For any κ-accessible comonad G and κ-cosignature $B : \mathcal{N}_\kappa \to A$ there is the following isomorphism which is natural in G and B:

$$[\mathcal{N}_\kappa, \mathcal{A}](V_\kappa G, B) \cong \mathbf{ACom}(\mathcal{A})(G, R_\kappa B) \tag{9}$$

Proof. The isomorphism is shown by a chain of natural isomorphisms provided by the two adjunctions in (6) and the full and faithful embedding of $[\mathcal{A}, \mathcal{A}]_\kappa$ into $\mathbf{AEnd}(\mathcal{A})$:

$$\begin{aligned}
[\mathcal{N}_\kappa, \mathcal{A}](V_\kappa G, B) &\cong [\mathcal{N}_\kappa, \mathcal{A}](_ \circ J_\kappa VG, B) \\
&\cong [\mathcal{A}_\kappa, A](VG, \mathrm{Ran}_{J_\kappa} B) \\
&\cong \mathbf{AEnd}(\mathcal{A})(VG, \mathrm{Ran}_{J_\kappa} B) \\
&\cong \mathbf{ACom}(\mathcal{A})(G, R\mathrm{Ran}_{J_\kappa} B) \\
&\cong \mathbf{ACom}(\mathcal{A})(G, R_\kappa B)
\end{aligned}$$

□

So given a κ-cosignature B we have constructed its representing comonad $R_\kappa B$. By Lemma 5.6, the category $R_\kappa B$–**Coalg** is isomorphic to the category $\mathrm{Lan}_{I_\kappa}\mathrm{Ran}_{J_\kappa} B$–**coalg**. Restricting ourselves to coalgebras over κ-presentable objects, the coalgebras over κ-presentable objects of the representing comonad $R_\kappa B$ are isomorphic to the κ-presentable models of the cosignature B as defined earlier. This is our partial dualisation of the result that the models of a signature are isomorphic to the Eilenberg-Moore category of the representing monad. Recall that a more complete characterisation is not possible because of the interplay of the left and right Kan extensions.

Unfortunately, the concrete construction of the representing comonad is rather involved in contrast to the dual, monadic case. There, the monad is constructed as a straightforward colimit of an chain of terms of increasing depth; here, the representing comonad for a κ-cosignature Σ is given in the proof of Lemma 5.7 by the Special Adjoint Functor Theorem, which in turn constructs it as the quotient of the generating set in the category of coalgebras of the cofree endofunctor over Σ. However, for particular endofunctors, this construction is well-known (Barr, 1993; Turi, 1996).

Example 5.10. The representing comonad $R_p \stackrel{def}{=} R_{\aleph_0}(B_p)$ for B_p is given by the cofree comonad on $G_p \stackrel{def}{=} \mathrm{Lan}_{I_\kappa}\mathrm{Ran}_{J_\kappa} B_p(X)$. $G_p(X)$ is $[[X, 2], 2]$ for X finite, and otherwise it is the union of $G_p(X_0)$ for all finite subsets $X_0 \subseteq X$. By Lemma 5.6, $R_p(X)$ is the final coalgebra of $X \times G_p$.
Now, $[X, 2] \cong \mathcal{P}_f(X)$ for X finite (where $\mathcal{P}_f$ is the finite powerset functor), and the final coalgebra for $X \times \mathcal{P}_f$ is well known (see e.g. (Turi, 1996, Chapter 13)): it is the set of infinite finitely branching trees with nodes labelled with elements from X, quotiented by the largest bisimulation. The same intuitions lead to the following characterisation of $R_p(X)$, which we owe to Dusko Pavlovic: We can consider the final $X \times G_P$ coalgebra as *bipartite, rooted, acyclic graphs* upto the appropriate notion of bisimulation. Concretely, these are potentially infinite trees whose nodes can have two colours, black and white say,

the root is black, the black nodes are labelled with elements from X, and all children of a black node are white nodes, and vice versa. Figure 1 shows three such bipartite trees.

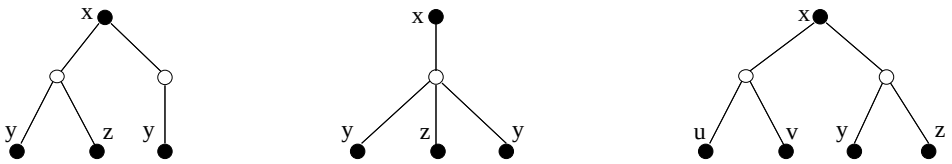


Fig. 1. Elements of $R_p(X)$, with $u, v, x, y, z \in X$

A (coalgebraic) *bisimulation* on these bipartite trees is a relation such that two trees t_1, t_2 are bisimilar iff:

— The colours of the root node of t_1 and the root node of t_2 are the same;
— if both are coloured black, then the labels of the root node of t_1 and the root node of t_2 are the same;
— if t_1' is a child of t_1, then there is a child t_2' of t_2 such that t_1' and t_2' are bisimilar;
— and if t_2' is a child of t_2, then there is a child t_1' of t_1 such that t_1' and t_2' are bisimilar.

For example, in Figure 1, the three trees can *not* be bisimilar. Note the definition is recursive, so the first condition is necessary even though the root of every element of the final coalgebra is always black.
The final coalgebra of $X \times G_p$, and hence the value of $B_p(X)$, is the set of all bipartite trees quotiented by the largest bisimulation R. With $[[X, 2], 2] \cong [\mathcal{P}_f(X), 2] \cong \mathcal{P}_f(\mathcal{P}_f(X))$, the coalgebra structure maps every such tree to a set of set of successors (hence the bipartite nature, i.e. the additional white layer).

Crucial for the development we have just given is that for finite X, $[X, 2] \cong \mathcal{P}_f(X)$, and maps $X \to [X, 2]$ can be considered as finitely branching graphs (or image-finite transition systems). Unfortunately, the same cannot be said for $[X, \omega]$, so the development of a similar syntax for the representing comonad of B_{p_ω} is still work in progress.

5.4. *Coequational Presentations and their Representing Comonad*

In this section, we continue the dualisation by defining *coequational presentations*, deriving a representing comonad for such a coequational presentation, and relating the coalgebras of the representing comonad to the models of the presentation. As we have seen above, equations are given as a pair of monad morphisms between free monads; the representing monad is defined to be the coequaliser of these. Dualising this gives a coequational presentation as a pair of comonad morphisms between cofree comonads, and takes the representing comonad as the equaliser of these comonad morphisms.

Definition 5.11 (Coequational Presentations). A *coequational presentation* is given by two cosignatures $B : \mathcal{N}_\kappa \to \mathcal{A}$ and $E : \mathcal{N}_\lambda \to \mathcal{A}$ (where $R_\kappa B$ is λ-accessible), and two comonad morphisms $\sigma, \tau : R_\kappa B \to R_\lambda E$.

Under isomorphism (9) the maps $\sigma, \tau : R_\kappa B \to R_\lambda E$ are given by maps $\sigma', \tau' : V_\lambda R_\kappa B \to E$, which in turn are families $\sigma'_c, \tau'_c : R_\kappa Bc \to Ec$ of maps for every $c \in \mathcal{N}_\lambda$. So, as opposed to the monadic case where an equation is given by a pair of terms, a coequation consists of two partitionings of the coterms; for example, in **Set** if $E(n) = \mathbf{2}$, each of σ and τ partition $R_\kappa B(n)$ into two subsets.

As mentioned above, given a coequational presentation, our intention is to define its representing comonad to be the equaliser of the comonad morphisms:

$$G \longrightarrow R_\kappa B \underset{\tau}{\overset{\sigma}{\rightrightarrows}} R_\lambda E \tag{10}$$

Proving that these equalisers exist is made easier by our abstract categorical setting which provides us with an alternative definition of the coalgebras of a comonad. Recall that such coalgebras are given by an object $x \in \mathcal{A}$ and a map $\alpha : x \to Gx$ which commutes with the unit and counit of G. First, observe that an object x of $\mathcal{A}$ is given by a map $X : \mathbf{1} \to \mathcal{A}$ (where $\mathbf{1}$ is the one-object category). Further, the functor category $[\mathbf{1}, \mathcal{A}]$ is isomorphic to $\mathcal{A}$, and we have

$$\mathcal{A}(x, Gx) \cong [\mathbf{1}, \mathcal{A}](X, G \circ X) \cong [\mathcal{A}, \mathcal{A}](\mathrm{Lan}_X X, G), \tag{11}$$

so to give a map $x \to Gx$ is equivalent to giving a natural transformation from $\mathrm{Lan}_X X \Rightarrow G$. In fact, $\mathrm{Lan}_X X$ is a comonad:

Lemma 5.12. $\mathrm{Lan}_X X$ is a comonad. If X is κ-presentable, then $\mathrm{Lan}_X X$ is κ-accessible.

Proof. Using the standard formula for left Kan extensions, $\mathrm{Lan}_X X(a) = \mathcal{A}(X, a) \otimes X$ (where $\otimes$ is the *tensor*; e.g. in **Set**, it is given by the usual product). If X is κ-presentable, then $\mathcal{A}(X, -)$ preserves all κ-filtered colimits, as does $- \otimes X$; hence $\mathrm{Lan}_X X$ is κ-accessible. To have a comonad structure, we need natural transformations $\mathrm{Lan}_X X \Rightarrow 1$ and $\mathrm{Lan}_X X \Rightarrow \mathrm{Lan}_X X \circ \mathrm{Lan}_X X$ which satisfy the comonad laws. The first of these is given by the image of the identity transformation on X under the isomorphism

$$[\mathbf{1}, \mathcal{A}](X, X) \cong [\mathcal{A}, \mathcal{A}](\mathrm{Lan}_X X, 1_A).$$

The second is given by the image of the transformation

$$X \overset{\varepsilon}{\Longrightarrow} \mathrm{Lan}_X X \circ X \overset{\mathrm{Lan}_X X \varepsilon}{\Longrightarrow} \mathrm{Lan}_X X \circ \mathrm{Lan}_X X \circ X$$

(where ε is the canonical transformation $X \Rightarrow \mathrm{Lan}_X X \circ X$) under the isomorphism

$$[\mathbf{1}, \mathcal{A}](X, \mathrm{Lan}_X X \circ \mathrm{Lan}_X X \circ X) \cong [\mathcal{A}, \mathcal{A}](\mathrm{Lan}_X X, \mathrm{Lan}_X X \circ \mathrm{Lan}_X X).$$

That the counit and comultiplication obey the comonad laws is easily verified. □

The fact that $\mathrm{Lan}_X X$ is a comonad allows us to strengthen equation (11) to obtain the promised characterisation of the coalgebras of a comonad.

Proposition 5.13. A coalgebra for a comonad G has a carrier object x in $\mathcal{A}$, given by a map $X : \mathbf{1} \to \mathcal{A}$, and a structure map $\alpha : x \to Gx$, given by a comonad morphism $\alpha' : \mathrm{Lan}_X X \Rightarrow G$ in $\mathbf{ACom}(\mathcal{A})$.

Proof. We have already seen above how under isomorphism (11) a structure map $\alpha' : x \to Gx$ corresponds to a natural transformation $\alpha : \mathrm{Lan}_X X \Rightarrow G$. It is then routine to verify that the properties of the structure map of a coalgebra correspond to the laws of a comonad morphism. □

With Proposition 5.13, we can show the existence of equalisers:

Lemma 5.14. The category of accessible comonads $\mathbf{ACom}(\mathcal{A})$ has all equalisers.

Proof. Consider two comonad morphisms $M \overset{\sigma}{\underset{\tau}{\rightrightarrows}} E$. We define the category (M,E)–**Coalg** to be given by coalgebras $\alpha : \mathrm{Lan}_X X \to M$ of M such that $\sigma \cdot \alpha = \tau \cdot \alpha$. Recall from the proof of Lemma 5.4 how we constructed the category of coalgebras as a weighted limit; using a similar diagram called an *equifier* (Kelly, 1989), we can construct (M,E)–**Coalg** as a weighted limit as well, making it accessible. With (M,E)–**Coalg**, M and E accessible, the forgetful functor from (M,E)–**Coalg** to $\mathcal{A}$ has a right adjoint; and post-composition of this right adjoint with the forgetful functor gives the equaliser of M and E. □

We finish this section by relating the models of a coequational presentation to the coalgebras of its representing comonad. If G is the representing comonad of the coequational presentation $\sigma, \tau : R_\kappa B \to R_\lambda E$, then a G-coalgebra is a map $\alpha : \mathrm{Lan}_X X \to G$ which by the universal property of equalisers is a map $\alpha' : \mathrm{Lan}_X X \to R_\kappa B$ such that $\sigma \cdot \alpha' = \tau \cdot \alpha'$, or in other words a coalgebra for B which equalises σ and τ. In this case, we say that the coalgebra $\mathrm{Lan}_X X \to R_\kappa B$ satisfies the coequational presentation $\sigma, \tau : R_\kappa B \to R_\lambda E$. Following the calculations above, if X is κ-presentable the map $\alpha : \mathrm{Lan}_X X \to R_\kappa B$, gives us two $R_\lambda E$-coalgebras, which by Lemma 5.4 are families $\sigma'' : \mathcal{A}(X,c) \to \mathcal{A}(X, Ec)$ and $\tau'' : \mathcal{A}(X,c) \to \mathcal{A}(X, Ec)$ of maps for all $c \in \mathcal{N}_\lambda$, which have to be equal if $\alpha : \mathrm{Lan}_X X \to R_\kappa B$ satisfies the presentation $\sigma, \tau : R_\kappa B \to R_\lambda E$.

5.5. *Examples of Functorial Coequational Presentations*

As we have seen in Example 5.10, the representing comonad even for a rather simple example can be unwieldy. But in order to consider coequations, we can just use the adjunction $R \vdash V$ between accessible endofunctors and accessible comonads. That is, we can define coequational presentations based on endofunctors:

Definition 5.15 (Functorial Coequational Presentations). A *functorial coequational presentation* is given by two accessible endofunctors $B : \mathcal{A} \to \mathcal{A}, E : \mathcal{A} \to \mathcal{A}$, and two comonad morphisms $\sigma, \tau : RB \to RE$.

The comonad represented by a functorial coequational presentation (we will drop the functorial in the following, relying on the context for the difference) is, of course, the equaliser of σ and τ in $\mathbf{ACom}(\mathcal{A})$ (which exists by Lemma 5.14). Also note that due to the adjunction $R \vdash V$, σ and τ can be more conveniently given as natural transformations $\sigma_X, \tau_X : VRBX \to EX$. Further, rather than figuring out the actual comonad it is more instructive, following the last paragraph of Sect. 5.4, to consider its coalgebras.

A simple example of a functorial coequational presentation in $\mathcal{A} = \mathbf{Set}$ is given by letting $BX = L \times X$ for a fixed set L of labels, and $EX = X$. Obviously, $RBX = \nu Y.X \times L \times Y$, which are streams of pairs $(\langle l_i, x_i \rangle)_{i \in \mathbb{N}}$ where $l_i \in L$ and $x_i \in X$. Now, define $\sigma_X(s) = x_0$ and $\tau_X(s) = x_2$ for $s \in VRBX$ (note that we map the whole stream to just a single result, since the maps are defined on the underlying functor $VRBX$). Concretely, a coalgebra $\alpha : A \to BA$ is a carrier set A, the elements of which we can consider as states, and a structure map α which for any state $a \in A$ returns a pair $\langle l, a' \rangle$, where $l \in L$, and a' is a successor state. (A, α) also defines a RB-coalgebra by unfolding the stream generated by each $a \in A$ to a sequence $S_a = \langle l_i, a_i \rangle_{i \in \mathbb{N}}$. This induced coalgebra is a coalgebra of the comonad given by the coequational presentation only if τ_X and σ_X agree on S, which concretely means that $a_2 = a$ and hence for all i, $a_i = a_{i+2}$. In other words, the states must be alternating and hence so must the labels since they are determined by the states. Note that all of these examples specify *equations* on streams. It is not possible to specify inequalities (e.g. $a_i \neq a_{i+1}$) using coequational presentations.

Now consider the functors $BX \stackrel{def}{=} \mathcal{P}_f X$ and $EX \stackrel{def}{=} \{tt, ff\}$. The elements of RBX are (possibly infinite) finitely branching trees with nodes labelled in X, quotiented by the largest bisimulation. The maps $\sigma_X, \tau_X : VRBX \to EX$ can be given as maps on the trees which are well-defined wrt. to the bisimulation. Let σ_X map every tree t to tt, while τ_X map a tree to tt iff it has at least one child. Then we specify infinite trees upto bisimulation.

Now let L be a set of labels, $BX \stackrel{def}{=} \mathcal{P}_f(L \times X)$ and $EX \stackrel{def}{=} \mathcal{P}_f L$. Recall that a coalgebra for B is a finitely branching transition system, or image-finite transition system. The final coalgebra of $X \times B_$ contains (equivalence classes of) trees with nodes labelled in X and edges labelled in L (this is close to what Turi (1996) calls an observational comonad). Let σ_X map a tree to the set of labels $l \in L$ of the outgoing edges from the root, and τ_X to a constant set of labels $L_0 = \{l_1, \ldots, l_n\}$. Then this specifies trees (upto bisimulation) such that, at every node, the set of labels issuing from the node is exactly the set L_0.

A final example would be to set $EX = \{tt, ff\}$, with $\sigma_X(t)$ is tt if there are no more than n outgoing edges (upto bisimulation), and ff otherwise, and $\tau_X(t)$ returns always tt. This would specify the comonad of n-ary transition systems, i.e. transition systems with no more than n outgoing branches from a given state.

5.6. *Examples of Coequational Presentations*

We can now combine the previous section with Example 5.10 to obtain a coequational presentation. Recall from Example 5.10 that the representing comonad R_p of the sig-

nature B_p maps a set X to equivalence classes of bipartite trees labelled with X. According to Definition 5.11, a coequational presentation is given by another cosignature $E : \mathbb{N} \to \mathbf{Set}$ and two natural transformations $\sigma, \tau : R_p \to R_\lambda E$, which can be more conveniently given as families $\sigma_c, \tau_c : V_\lambda R_p(c) \to E(c)$ for $c \in \mathbb{N}$. The underlying co-signature $V_\lambda R_p : \mathbb{N} \to \mathbf{Set}$ of R_p maps a natural number c to $R_p(c)$.

The examples from the previous section can be rephrased in this setting, but of course the corresponding endofunctor is the underlying endofunctor of R_p, the representing comonad: we can set $E(n) = \{tt, ff\}$ and define $\sigma_n(t) = tt$ for all t and $\tau_n(t) = tt$ if t has at least one node. This then specifies the infinite bipartite graphs. For another example, set $\sigma_n(t) = tt$ if the top of t is (upto bisimulation) the rightmost tree in Figure 1 and $\sigma_n(t) = ff$ for all other t. Set $\tau_n(t) = tt$ for all t. Then the represented comonad has all coalgebras of R_p except those which as a transition system can reach four different states after the second iteration, since for these $\sigma_n(A) \neq \tau_n(A)$.

As we can see, functorial equational presentations allow us to specify coalgebraic structures in a very flexible way. Unfortunately, due to the unwieldy structure of the representing comonad, coequational presentations based on signatures are not that flexible. Also, we have made light of the fact that the elements of the cofree comonad on B_p are equivalence classes up to bisimulation, which makes a precise syntactical description clumsy; we assume this will not improve for cosignatures like B_{p_ω} (Example 5.2).

Finally, the precise connections of coequational presentations to the predicate transformer semantics sketched in Section 5.2 remains to be investigated, in particular whether we can give the semantics for a programming language in terms of coequations, or if we can specify programs by means of coequations.

6. Conclusions, Related and Further Work

This paper set out to analyse the essence of the categorical approach to universal algebra based upon the theory of accessible monads and then derive their coalgebraic counterparts. While there is potential for further development, we feel we have made a definite contribution which will be of interest to the general coalgebra community. In particular, the derivation of the infinitary monad corresponding to the collection of final $X + F$-coalgebras seems to have ready applications in lazy functional programming. In addition, the derivation of the cofree comonad representing a cosignature and the interpretation of a coequational presentation as an equaliser of cofree comonads are dualisations of the corresponding results in the standard theory. We have also related the coalgebras of the representing comonad to the models of the coequational presentations.

The major technical challenge was caused by the increase in rank when passing from a cosignature to its representing comonad. For example, this meant that the characterisation of coalgebras only holds for coalgebras up to a certain rank. In addition, this increase in rank makes it unlikely that every accessible comonad can be given as a coequational presentation (i.e. as an equaliser of cofree comonads), as is possible for finitary monads.

Of course there remains much to be done. Our approach has been very abstract and

perhaps the most important next step is to make precise the relationship between our approach and others pursued in the community.

For example, Cirstea (Cirstea, 2000) defines an *abstract cosignature* as a functor $F : \mathcal{C} \to \mathcal{C}$ where $\mathcal{C}$ has all finite limits and limits of ω^{op}-chains, and F preserves pullbacks and such limits. Then, an *observer* is given by a functor $K : \mathcal{C} \to \mathcal{C}$ and a natural transformation $c : U \to K{\cdot}U$, and a coequation by two observers (K, l) and (K, r). Note that the transformation c gives a functor $c^* : F\text{–}\mathbf{coalg} \to K\text{–}\mathbf{coalg}$ such that $U = U{\cdot}c^*$. Let RF and RK be the cofree comonads on F and K respectively, then $F\text{–}\mathbf{coalg} \cong RF\text{–}\mathbf{Coalg}$, and we have a functor $c^{\#} : RF\text{–}\mathbf{Coalg} \to RK\text{–}\mathbf{Coalg}$, which is equivalent to a comonad morphism $c^{\dagger} : RF \to RK$. Thus, her coequations give pairs of two such comonad morphisms, and are captured in our framework.

Recent work on co-Birkhoff theorems (Awodey and Hughes, 2000; Rosu, 2000; Kurz, 2000) uses stronger assumptions on the base category, and can thus prove stronger results in a more limited setting. For example, Awodey and Hughes (2000) characterise coequational covarieties in categories which are regularly well-powered, cocomplete, have epi-regular mono factorizations and enough injectives. Both coequational covarieties and comonads characterise categories of coalgebras; the former more concretely, due to the stronger assumptions on the base, the latter more abstract. An interesting approach has been taken by Adamek and Porst (2001) : they dualize the construction of the free algebra sequence by a transfinite cofree coalgebra construction, and define a *covarietor* to be a functor for which the cofree coalgebra exists (i.e. the coalgebra construction converges at some point). Recall that in our case, the dualization of the free algebra sequence fails because of the mixture of left and right Kan extensions and hence we are left with the less descriptive construction of the cofree coalgebra in Lemma 5.7. In (Awodey and Hughes, 2000; Adamek and Porst, 2001), coequations are defined as subobjects of, or monomorphism into, the cofree coalgebra; a coalgebra satisfies the coequation if the arrow into cofree coalgebra filters through the coequation. The relation to a coequational presentation is obvious: the universal property of the equaliser says exactly that it is a morphism into the cofree comonad (i.e. the cofree coalgebra) such that every other coalgebra equalising the two arrows filters through the equaliser, and further, every equaliser is a monomorphism. Thus, the two approaches are in principle equivalent, but the intent there was to characterise the dual of varieties, whereas our goal was to characterise the dual of the equational presentation of monads.

We have not as yet developed the logical calculus underlying the categorical constructions of the comonadic semantics. It seems this underlying logic will have a modal flavour, and hence research in this area, e.g. by Kurz (2000) , seems relevant.

We would like to thank Hans-Eberhard Porst for pointing out a minor error in an earlier version of the paper, Micheal Barr, Dusko Pavlovic and the categories mailing list for helping us to figure out the structure of the final coalgebra in Example 5.10, and the anonymous referees for their constructive criticism.

References

Aczel, P., Adamek, J., and Velebil, J. (2001). Iteration monads. In Montanari, U., editor, *Proceedings CMCS'01*. Electronic Notes in Theoretical Computer Science 44.

Adamek, J. (2000). On final coalgebras of continuous functors. To appear.

Adamek, J. and Porst, H.-E. (2001). From varieties of algebras to covarieties of coalgebras. In Montanari, U., editor, *Proceedings CMCS'01*. Electronic Notes in Theoretical Computer Science 44.

Adamek, J. and Rosický, J. (1994). *Locally Presentable and Accessible Categories*. Number 189 in London Mathematical Society Lecture Note Series. Cambridge University Press.

Awodey, S. and Hughes, J. (2000). The coalgebraic dual of Birkhoffs variety theorem. `http://www.contrib.andrew.cmu.edu/user/jesse/papers/CoBirkhoff.ps.gz`.

Barr, M. (1993). Terminal algebras in well-founded set theory. *TCS*, 114:299– 315.

Barr, M. and Wells, C. (1985). *Toposes, Triples and Theories*. Springer Verlag.

Cirstea, C. (2000). An algebra-coalgebra framework for system specification. In *Proceedings CMCS 2000*, volume 33 of *Electronic Notes in Theoretical Computer Science*.

Dijkstra, E. W. (1976). *A Discipline of Programming*. Prentice-Hall.

Dubuc, E. J. and Kelly, G. M. (1983). A presentation of topoi as algebraic relative to categories or graphs. *Journal for Algebra*, 81:420–433.

Fiore, M., Plotkin, G., and Turi, D. (1999). Abstract syntax and variable binding. In *Proc. LICS'99*, pages 193– 202. IEEE Computer Society Press.

Hutton, G. and Gibbons, J. (2001). The generic approximation lemma. *Information Processing Letters*, 79(4):197– 201.

Johnstone, P. T., Power, A. J., Tsuijiushita, T., Watanabe, H., and Worrell, J. (1998). The structure of categories of coalgebras. To appear in *TCS*.

Kelly, G. M. (1980). A unified treatment of transfinite constructions for free algebras, free monoids, colimits, associated sheaves and so on. *Bull. Austral. Math. Soc.*, 22:1– 83.

Kelly, G. M. (1989). Elementary observations on 2-categorical limits. *Bull. Austral. Math. Soc.*, 39:301–317.

Kelly, G. M. and Power, A. J. (1993). Adjunctions whose counits are coequalizers, and presentations of finitary monads. *Journal for Pure and Applied Algebra*, 89:163– 179.

Kurz, A. (2000). *Logics for Coalgebra and Applications to Computer Science*. Dissertation, Ludwig-Maximilans-Universtität München.

Lüth, C. (1996). Compositional term rewriting: An algebraic proof of Toyama's theorem. In *Rewriting Techniques and Applications RTA'96*, LNCS 1103, pages 261– 275. Springer Verlag.

Lüth, C. and Ghani, N. (1997). Monads and modular term rewriting. In *Category Theory in Computer Science CTCS'97*, LNCS 1290, pages 69– 86. Springer Verlag.

Mac Lane, S. (1971). *Categories for the Working Mathematician*. Springer Verlag.

Makkai, M. and Paré, R. (1989). *Accessible Categories: The Foundations of Categorical Model Theory*, volume 104 of *Contemporary Mathematics*. American Mathematical Society.

Moss, L. (1999). Coalgebraic logic. *Annals of Pure and Applied Logic*, 96:277–317.

Robinson, E. (1994). Variations on algebra: monadicity and generalisations of equational theories. Technical Report 6/94, Sussex Computer Science Technical Report.

Rosu, G. (2000). Equational axiomatisability for coalgebra. *TCS*, 260(1).

Turi, D. (1996). *Functorial Operational Semantics and its Denotational Dual*. PhD thesis, Free University, Amsterdam.

Winskel, G., editor (1993). *The Formal Semantics of Programming Languages*. MIT Press.

Worrell, J. (1999). Terminal sequences for accessible endofunctors. In *Proceedings CMCS '99*, volume 19. Electronic Notes in Theoretical Computer Science 19.