 Electronic Notes in Theoretical Computer Science 203 (2008) 175–194 

[www.elsevier.com/locate/entcs](http://www.elsevier.com/locate/entcs)

Functorial Coalgebraic Logic: The Case of Many-sorted Varieties

Alexander Kurz

*Department of Computer Science University of Leicester, UK*

Daniela Petri¸san

*Department of Computer Science University of Leicester, UK*

**Abstract**

Following earlier work, a modal logic for *T* -coalgebras is a functor *L* on a suitable variety. Syntax and proof system of the logic are given by presentations of the functor. This paper makes two contributions. First, a previous result characterizing those functors that have presentations is generalized from endofunctors on one-sorted varieties to functors between many-sorted varieties. This yields an equational logic for the presheaf semantics of higher-order abstract syntax. As another application, we show how the move to functors between many-sorted varieties allows to modularly combine syntax and proof systems of different logics. Second, we show how to associate to any set-functor *T* a complete (finitary) logic *L* consisting of modal operators and Boolean connectives.

*Keywords:* Coalgebra, Modal Logic, Stone Duality, Coalgebraic Logic, Sifted Colimits, Variety, Universal Algebra, Presentation by Operations and Equations

# Introduction

Beginning with [[6](#_bookmark28),[17](#_bookmark39)], it has been argued that logics for *T* -coalgebras (where *T* is an endofunctor on Set) are suitably described by endofunctors *L* on the category of Boolean algebras. Syntactically, *L* specifies an extension of Boolean propositional logic by modal operators and axioms. Semantically, *L* gives a logical description of the ‘transition type’ *T* of the coalgebras.

[[17](#_bookmark39)] showed that the modal logics for coalgebras usually considered [[22](#_bookmark42),[26](#_bookmark43),[15](#_bookmark37)] give rise to such functors *L*. [1](#_bookmark0) To describe the class of functors arising in such a way,

[[8](#_bookmark29)] introduced the notion of a functor having a finitary presentation by operations and equations. This notion was investigated more systematically in [[20](#_bookmark44)] where it is

1 Although quite different from the above, Moss’s coalgebraic logic [[21](#_bookmark45)] also fits into our framework.

1571-0661 © 2008 Elsevier B.V. Open access under [CC BY-NC-ND license.](http://creativecommons.org/licenses/by-nc-nd/3.0/)

doi:10.1016/j.entcs.2008.05.025

shown that a functor *L* on a (finitary) variety *A* has a presentation by operations and equations if and only if *L* preserves sifted colimits. Although not as well known as filtered colimits, sifted colimits are the right concept when working with varieties (as opposed to locally finitely presentable categories): Each variety *A* is the free cocompletion by sifted colimits of the dual of the Lawvere theory of *A*. The reason is that algebras for a Lawvere theory are set-valued product-preserving functors and sifted colimits are precisely those colimits that commute in Set with finite products.

This paper continues this line of research. We start by generalizing the results of

[[20](#_bookmark44)] on functors on varieties from the one-sorted to the many sorted case (Section [3](#_bookmark3)). This generalization in itself is not difficult, but it has interesting applications. The first, maybe somewhat unexpected, is that it provides an equational logic for the binding algebras used in the work of [[11](#_bookmark33)] (Section [4](#_bookmark14)). The second application shows how to modularly compose presentations of different functors (Section [5](#_bookmark16)). Even if one was only interested in one-sorted coalgebras, only the move to many-sorted logics gives the desired modularity. Finally, whereas [[20](#_bookmark44)] shows how to prove strong completeness results by generalizing the J´onsson-Tarski representation theorem for Boolean algebras with operators, we show here how to prove completeness results for categories of coalgebras that do not satisfy the restrictions needed for the strong completeness result of [[20](#_bookmark44)]. In particular, we show that an arbitrary set-functor has a complete finitary logic (Section [6](#_bookmark18)).

**Acknowledgments** We would like to thank Rick Thomas for pointing out a very useful reference [[5](#_bookmark26)].

# Some preliminaries

For an endofunctor *L* on a category *A*, we consider the category of *L-algebras*, denoted by Alg(*L*), whose objects are defined as pairs (*A, α*) such that *α* : *LA → A* is a morphism in *A*. A morphism of *L−*algebras *f* : (*A, α*) *→* (*A', α'*) is a morphism *f* : *A → A'* of *A* such that *f ◦α* = *α' ◦Lf* . Dually, for an endofunctor *T* : *A→A* we consider the category of *T−coalgebras*, denoted by Coalg(*T* ), whose objects are pairs (*A, γ*), such that *γ* : *A → T A*. A morphism of *T−*coalgebras *f* : (*A, γ*) *→* (*A', γ'*) is an arrow *f* : *A → A'* of *A* such that *Tf ◦ γ* = *γ' ◦ f* .

Let *S* be a set (of sorts). A *signature* Σ is a set of operation symbols together with an arity map *a* : Σ *→ S∗ × S* which assigns to each element *σ ∈* Σ a pair (*s*1*,..., sn*; *s*) consisting of a finite word in the alphabet *S* indicating the sort of the arguments of *σ* and an element of *S* indicating the sort of the result of *σ*. To each signature we can associate an endofunctor on Set*S*, which will be denoted for simplicity with the same symbol Σ:

(Σ*X*)*s* =(

*k∈ωS*

*f*

Σ*k,s × Xk*)*s*

Here, by *ωS* we denote the set of functions from *S* to *ω* which have finite support

*f*

(i.e. which vanish everywhere except for a finite set) and by *Xk* the set of presheaf

morphisms Set*S*(*k, X*). In detail, if *k ∈ ωS* has support *{s*1*, ..., sn}* then Σ*k,s* is

*f*

a set of operations of arity (*s*1*...sn*; *s*) and *Xk* is isomorphic in Set with the finite

product *Xk*(*s*1) *× ··· × Xk*(*sn*). Conversely to each endofunctor on Set*S* given as

*s*1 *sn*

above corresponds a signature

*k∈ωS*

*f*

Σ*k,s*. Throughout this paper we will make no

notational difference between the signature and the corresponding functor, and it will be clear from the context when we refer to the set of operation symbols or to a Set*S* endofunctor. The algebras for a signature Σ are precisely the algebras for the corresponding endofunctor, and form the category denoted by Alg(Σ). The terms over an *S*-sorted set of variables *X* are defined in the standard manner and and form a set denoted by TermΣ(*X*), and in fact this is the underlying set of the free Σ-algebra generated by *X*. An equation consists of a pair (*τ*1*, τ*2) of terms of the same sort, usually denoted *τ*1 = *τ*2. A Σ-algebra *A* satisfies this equation if and only if, for any interpretation of the variables of *X*, we obtain equality in *A*. A class *A* of Σ-algebras is called a *variety* or an *equational class* if there exists a set of equations *E* such that an algebra lies in *A* if and only if it satisfies all the equations of *E*. Such an equational class will be denoted by Alg(Σ*, E*). The forgetful functor *U* : *A →* Set*S* has a left adjoint *F* and Alg(Σ*, E*) is monadic over Set*S*, i.e it is isomorphic with the Eilenberg-Moore category (Set*S*)*T* for the monad *T* corresponding to the adjoint pair *F E U* . Also notice that *UF* is a finitary, that is, *UF* preserves filtered colimits. In [[3](#_bookmark23)], varieties are described independently of signatures and equations, via finite product sketches, and it is proved that any variety is a locally finitely presentable category. Recall that a category is locally finitely presentable provided that it is cocomplete and it has a set *K* of finitely presentable objects such that any object is a filtered colimit of objects of *K*. We will say that a category is *lfp* if it is locally finitely presentable.

Endofunctors may appear via composition of functors between different varieties. Therefore, it is useful to consider a slight generalization of the notion of signature. If *S*1 and *S*2 are sets of sorts we will consider operations with arguments of sorts in *S*1 and returning a result of a sort in *S*2, encompassed in the signature functor Σ: Set*S*1 *→* Set*S*2 :

2

Σ*X* =(

*k∈ωS*1

*f*

Σ*k,s × Xk*)*s∈S*

(1)

An important example of a (finitary) variety of algebras is the functor category Set*C* for any small category *C*. The sorts are the objects of *C*, the operations symbols are the morphisms of *C* (all of them with arity 1), and the equations are given by the composition of morphisms in *C*.

A *sifted category D* is a small category such that colimits over *D* commute in Set with finite products. A sifted colimit in a category *C* is a colimit over *D*. The most important examples of sifted colimits are filtered colimits and reflexive coequalizers.

An object in a category is called *strongly ﬁnitely presentable* if its hom-functor preserves sifted colimits. It is shown in [[4](#_bookmark27)] that any object in a variety is a sifted colimit of strongly finitely presentable algebras, which in a variety are the retracts of finitely generated free algebras. An important observation is that sifted colimit preserving functors on varieties are determined by their action on free algebras. Another useful observation is that for an *S*-sorted variety *A* the forgetful functor *U* : *A →* Set*S* and its left adjoint *F* are sifted colimit preserving functors. In fact *F* preserves all colimits.

# Presenting algebras and functors

The purpose of this section is an algebraic investigation of logics for coalgebras. The general idea is as follows. Just as coalgebras are given wrt a functor *T* on, say, Set, so are logics for coalgebras given by a functor *L* on, say, Boolean algebras. The following example shows how logics for coalgebras given in a more conventional style give rise to a functor on the category BA of Boolean algebras.

**Example 3.1** Let *T* = *У* be the covariant powerset functor. The modal logic **K** associated to *У*-coalgebras (=Kripke frames) can be described by the functor *L* which maps a Boolean algebra *A* to the Boolean algebra *LA* freely generated by

*{ a | a ∈ A}* modulo the relations *T* = *T* and(*a Λ b*)= *a Λ b*. We see that the modal operators appear as generators and the modal axioms as relations. Of course, from a logical point of view, we want the generators to be operations and the relations to be universally quantified equations. In other words, we need that the description of *LA* in terms of generators and relations is uniform in *A*. This is exactly captured by Definition [3.3](#_bookmark5) below.

It is not difficult to see that the category Alg(*L*) of algebras for the functor *L* is isomorphic to the category of Boolean algebras with operators, which constitute the standard algebraic semantics of **K** in modal logic. In particular, the initial *L*-algebra is the Lindenbaum-Tarski algebra of the modal logic **K**.

To simply replace a concrete modal logic by the corresponding functor is a powerful abstraction that makes a number of category theoretic methods available to modal logic. This section makes sure that the move from logics to functors is not an over- generalisation: Every suitable functor *L* will come from a modal logic in exactly the same way as in the example above. The reader who wants to know more about the relationship between *T* -coalgebras and *L*-algebras before reading this section might want to skip ahead to Section [6](#_bookmark18).

**Definition 3.2** Let *A* be a many-sorted algebra in a variety *A*. We say that (*G, E*) is a presentation for *A* if *G* is an *S*-sorted set of generators and *E* = (*Es*)*s∈S, Es ⊂* (*UFG*)*s ×* (*UFG*)*s* is an *S* sorted set of equations such that *A* is the coequalizer of the following diagram:

*π q*

1 *F G A*  *A*

*FE*

*π*

2

(2)

The maps *π, π* are induced, via the adjunction, by the projections *π*1*, π*2 of *E* on

1 2

*UFG*.

Next we want to define a presentation for a functor *L* : *A*1 *→ A*2 between many- sorted varieties. For *i ∈ {*1*,* 2*}*, denote by *Si* the set of sorts for *Ai* respectively,

by *Ui*

*i*

: *Ai*

*→* Set*Si* the corresponding forgetful functor, and by *F*

its left adjoint.

We will do this in the same fashion as in [[20](#_bookmark44)] and [[8](#_bookmark29)], keeping in mind that we need to extend ([2](#_bookmark4)) uniformly: this means that the generators and equations for each *LA* will depend functorialy on *A*. Suppose *A* is a many-sorted algebra in *A*1. The generators Σ*U*1*A* for the algebra *LA* will be given by a signature functor Σ : Set*S*1 *→* Set*S*2 as in ([1](#_bookmark2)). The equations that we will consider are of rank 1, meaning that in the terms involved every variable is under the scope of precisely one operation symbol in Σ, and are given by an *S*2-sorted set *E*. For each sort

*s ∈ S*2 and each *S*1-sorted set of variables *V* with the property that *Vt* is finite,

*t∈S*1

we consider a set *EV,s* of equations over the set *V* of terms of sort *s*, which is defined

as a subset of (*U*2*F*2Σ*U*1*F*1*V* )2. Now take *EV* = (*EV,s*)*s∈S* and *E* = *EV* .

*s*

*V ∈ωS*1

2

*f*

**Definition 3.3** Let *S*1*, S*2 be sets of sorts, *A*1 be an *S*1-sorted variety and *A*2 be an *S*2-sorted variety. A presentation for a functor *L* : *A*1 *→ A*2 is a pair *⟨*Σ*, E⟩* defined as above. A functor *L* : *A*1 *→ A*2 is presented by *⟨*Σ*, E⟩*, if

* 1. for every algebra *A ∈ A*1 the algebra *LA* is the joint coequalizer:

*π*

1 *F*2Σ*U*1*v*  *qA*

*F E*  *F* Σ*U F V F* Σ*U A LA* (3)

2 *V* 2 1 1 2 1

*π*

2

taken after all finite sets of *S*1-sorted variables *V* and all valuations *v* : *V → U*1*A*. Here *v* denotes the adjoint transpose of a valuation *v*.

* 1. for all morphisms *f* : *A → B* the diagram commutes:

*F* Σ*U A*  *qA*  *LA*

2 1

*F*2Σ*U*1*f Lf*

(4)

*F* ΣJ *qB* J

2 *U*1*B LB*

The importance of this notion, emphasized in [[8](#_bookmark29)], resides in the fact that endo- functors having finitary presentations give rise to modal logics, where the modal operators are the operation symbols of Σ and the axioms are the equations of *E*.

If *A* = Alg(Σ*A, EA*) is an *S*-sorted variety and the endofunctor *L* : *A → A* has a finitary presentation *⟨*Σ*L, EL⟩*, we can obtain an equational calculus for Alg(*L*), regarding the equations *EA* and *EL* as equations containing terms in TermΣ*A*+Σ*L* . First remark, that formally, for an arbitrary set of variables *V* , *EL,V* is a subset of the

*S−*sorted set (*UF* Σ*LUFV* )2. But for each set *X*, *UFX* is a quotient of TermΣ *X*

*A*

modulo the equations. Thus, if we choose a representative for each equivalence class in *UF* Σ*LUFV* , we can obtain a set of equations in TermΣ*A* Σ*L*TermΣ*A* . Using the natural map from TermΣ*A* Σ*L*TermΣ*A* to TermΣ*A*+Σ*L V* , we obtain a set of equations on terms TermΣ*A*+Σ*L V* . By abuse of notation we will denote this set with *EL* as well.

**Theorem 3.4** *Let A* = Alg(Σ*A, EA*) *be an S-sorted variety and let L* : *A → A be a functor presented by operations* Σ*L and equations EL. Then* Alg(*L*) *∼*= Alg(Σ*A* + Σ*L, EA* + *EL*)*.*

**Proof.** We define a functor *H* : Alg(*L*) *→* Alg(Σ*A* + Σ*L, EA* + *EL*). Suppose *α* : *LA → A* is an *L*-algebra. Then the underlying set of *HA* is defined to be *UA*. *HA* inherits the algebraic structure of *A*: the interpretation of the operation symbols of Σ*A* is the same as in the algebra *A* and it satisfies the equations *EA*. As far as the operation symbols of Σ*L* are concerned, their interpretation is given by the composition:

*F* Σ *UA*  *qA*  *LA*  *α*  *A* (5) Explicitly, the interpretation of an operation symbol *σ* of arity (*s*1 *... sn*; *s*) is the

*L*

morphism *σA* : *As*

1

*×· · · × Asn*

*→ As* defined by

*σA*(*x*1*,..., xn*)= *α*(*qA*((*σ, x*1*,..., xn*)))

Now it is clear that the equations *EL* are satisfied in *HA*, because *qA* is a coequalizer as in ([3](#_bookmark6)). If *f* is a morphism of *L−*algebras, we define *Hf* = *f* and we only have to check that *f* (*σ*(*a*1*,..., ak*)) = *σ*(*f* (*a*1)*,...,f* (*ak*)) for all *σ ∈* Σ*L*. But this follows from the fact the definition of the interpretation of the operations, the commutativity of diagram ([4](#_bookmark7)) and the fact that *f* is an *L*-algebra morphism.

Conversely, we define a functor *J* : Alg(Σ*A* + Σ*L, EA* + *EL*) *→* Alg(*L*). Suppose *A*

is an algebra in Alg(Σ*A* + Σ*L, EA* + *EL*). The map *ρA* : Σ*LUA → UA* defined by:

induces a map *ρ*

(*σ*(*s*1*...sn*;*s*)*, xi*1 *,..., xin* ) *'→ σ*(*s*1*...sn*;*s*)(*xi*1 *,..., xin* )

: *F* Σ*LUA → A*. The fact that equations *EL* are satisfied implies

*A*

that *ρ*

*A*

* *F* Σ*LUv ◦ π*

*A*

1

= *ρ*

* *F* Σ*LUv ◦ π*

as depicted in ([6](#_bookmark9)). But *LA* is a

coequalizer in Alg(Σ*A, EA*), therefore there exists a morphism *αA* : *LA → A* such that *αA ◦ qA* = *ρ* . We define *JA* to be the *L*- algebra *αA*. For any morphism *f* : *A → B* in Alg(Σ*A* + Σ*L, EA* + *EL*) we define *Jf* = *U*0*f* , where *U*0 : Alg(Σ*A* + Σ*L, EA* + *EL*) *→* Alg(Σ*A, EA*) is the forgetful functor. This is well defined and we can check this easily by proving that the rightmost square of diagram ([6](#_bookmark9)) is commutative:

2

*A*

*π*

1

*F* Σ*Uv ,F* Σ*¸L*

*,,*

*,,,,*1*, ,*

*A*

*UA*  *qA*  *LA \_ \_α\_A \_ \_*z *A˛*

*ρ*

*FEL*

*F* Σ*LUF V¸*

*F* Σ*Uf*

*Lf f*

(6)

*¸¸F¸¸*Σ*¸Uv*2

*π*

*¸¸¸*

2

z*˛*J



*ρ*

*B*

JJ

*F* Σ*L*

*UB*  *qB*  *LB \_ \_α\_B \_ \_* *B ¸*

Now it is straightforward to check that *J ◦ H* and *H ◦ J* are the identities.

The characterization of endofunctors having finitary presentation was given in [[20](#_bookmark44)] for monadic categories over Set and it can be easily extended if we replace Set with the presheaf category Set*S*. The result holds even if we work with functors between different varieties.

**Theorem 3.5** *Let S*1*, S*2 *be sets of sorts, A*1 *be an S*1*-sorted variety and A*2 *be an S*2*-sorted variety. For a functor L* : *A*1 *→ A*2 *the following conditions are equivalent:*

1. *L has a ﬁnitary presentation by operations and equations;*
2. *L preserves sifted colimits.*

**Proof.** (*i*) *⇒* (*ii*). Assume *L* has a finitary presentation *⟨*Σ*, E⟩*. Let *D* be a sifted category and *ai* : *Ai → A* be a sifted colimit in *A*1. Let *di* : *LAi → B* be an arbitrary cocone. As we have seen in the preliminaries, the corresponding forgetful functors and their left adjoints *U*1*, U*2*, F*1*, F*2 preserve sifted colimits. Σ shares the same property because sifted colimits are computed point-wise and commute with finite products. Therefore we obtain that *F*2Σ*U*1*ai* : *F*2Σ*U*1*Ai → F*2Σ*U*1*A* is a colimiting cocone in *A*2, hence there exists a map *d* : *F*2Σ*U*1*A → B* such that *d ◦ F*2Σ*U*1*ai* = *di ◦ qAi* for all *i* in *D*.

Choose an arbitrary *S*1-sorted set of variables *V* = (*Vs*)*s∈S*1 such that *Vs* is finite

*s∈S*1

and a morphism *v* : *V → U*1*A*. Then *V* is strongly finitely presented in the category Set*S*1 , and *U*1*ai* : *U*1*Ai → U*1*A* is a sifted colimit, so there exists *vi* : *V → U*1*Ai* such that *v* = *U*1*ai ◦ vi*. From the fact that *qAi* is a joint coequalizer, it follows that *d* makes the bottom line of diagram ([7](#_bookmark11)) commutative.

*F*2Σ¸*U,*1*¸Ai*

*sss*

*ss*

*¸¸*

*F*2Σ*U*1*vsiss*

*¸¸¸*

*F* Σ*U a ¸¸d¸i◦qAi*

*¸¸*

*sss*

*ss*

*π sss*

2 1 *i*

J

*¸¸¸*

*¸¸¸*

(7)

*F E*  1 *F*2Σ*U*1*v*  *d* z

*F* Σ*U F V F* Σ*U A*

2 *V π*2 2 1 1 2 1 *B*

Using that *LA* is a joint coequalizer we obtain *b* : *LA → B* such that *b ◦ qA* = *d*. Now it is immediate to check that diagram ([8](#_bookmark12)) is commutative, and this shows that

the cocone *Lai* : *LAi → LA* is universal.

*F*2Σ*U*1*Ai*

*qAi*  *LA*

*,,,,*

*,*

*i*

*F*2Σ*U*1*Ai*

*Lai*

*,,,*

*F* ΣJ *qA*  J

*,,,di*

(8)

2 *U*1*A¸¸¸¸¸ LA¸¸¸ ,,,*

*¸¸¸¸¸¸d¸¸ ¸¸¸¸b¸ ,,,*

*¸¸¸¸¸¸ ¸¸¸,,,*

*¸¸¸¸¸¸*zzz *\_z*

*¸*

*B*

(*ii*) *⇒* (*i*) Being a sifted colimit preserving functor, *L* is determined by its values

on finitely generated free algebras. Given *k ∈ ωS*1 with support *{s*1*,..., sn}* and

*f*

given *s ∈ S*2 we can view the elements of the set (*U*2*LF*1*k*)*s* as operations symbols which take *k*(*si*) arguments of sort *si* for all 1 *≤ i ≤ n* and return a result of sort *s*. More explicitly we can consider for all algebras *A* the map *rA* given component-wise by:

(*U LF k*)

*× U Ak rA,s*

(*U LA*)

(9)

2 1 *s*

*k∈ωS*

*f*

1 *−−−−−−→* 2 *s*

(*σ, x*) *'→* (*U*2*LϵA ◦ U*2*LF*1*x*)*s*(*σ*)

where *ϵA* : *F*1*U*1*A → A* is the counit of the adjunction. In the definition of the map *rA,s* we have interpreted *x* as a morphism in Set*S*(*k, U*1*A*). Now the operations that we will consider are encompassed in the functor Σ : Set*S*1 *→* Set*S*2 defined by

2

Σ*X* =(

*k∈ωS*1

*f*

(*U*2*LF*1*k*)*s × Xk*)*s∈S*

(10)

Note that *r* is a natural transformation from Σ*U*1 to *U*2*L*.

For an arbitrary *S*1-sorted set of variables *V* , the equations are induced by the map *rF*1*V* : Σ*U*1*F*1*V → U*2*LF*1*V* as in ([9](#_bookmark13)), more precisely *EV* is defined to be the

kernel pair of the map *Ur*

*F V*

1

: *U*2*F*2Σ*U*1*F*1*V → U*2*LF*1*V* . We will prove that *L* is

presented by *⟨*Σ*, E⟩*. For all *k ∈ ωS*1 the following diagram is a split coequalizer

*f*

because *Ek* is a kernel pair.

*π*1

*E*

*U*2*r*

Σ*U F k F*1*k*  *U LF k*



,*¸*

*π*2

*U*2 *F*

(11)

*k*

*t*

2 1 ¸1 *\_ * 2 1

*s*

One can check that it follows that

*U*2*π U*2*r*

1 *F*1*k*

*U F E*



¸*k ¸*

*U F* Σ*U F k U LF k*

(12)

2 2

*U*2*π*

2 2 1 ¸1 *\_* 2 1

*s*

2

*U*2*F*2*t ◦ ηU*2*F*2Σ*U*1*F*1*k*

is again a split coequalizer. *U*2 is a monadic functor, hence it creates split coequal- izers, and we obtain that

*π r*

1 *F*1*k*

*F E*  *F* Σ*U F k LF k*

(13)

2 *k* 2 1 1 1

*π*

2

is a coequalizer. Now it is straightforward to show that

*π r*

*F E*  1 *F*2Σ*U*1*v*

*F* Σ*U F V F* Σ

*F* 1*k*  *LF k*

(14)

2 *V* 2 1 1

*π*

2

2 *U*1*F*1*k* 1

is a joint coequalizer. This proves that *L* coincides on finitely generated algebras with the functor presented by the finitary presentation *⟨*Σ*, E⟩*, and therefore it is presented by *⟨*Σ*, E⟩*.

# Equational logic for higher-order abstract syntax

Syntax with variable binders cannot be captured as an initial algebra in the usual way. But Fiore, Plotkin and Turi [[11](#_bookmark33)] (see also Hofmann [[13](#_bookmark35)] and Gabbay and Pitts [[12](#_bookmark34)]) showed that this is possible if one moves from algebras for a functor on Set to algebras for a functor on a suitable presheaf category. In particular, they showed that *λ*-terms up to *α*-equivalence form an initial algebra for a functor. These functors generalize the notion of a signature, but a notion of equational theory for these algebras is missing in [[11](#_bookmark33)] (but see the more recent work [[10](#_bookmark32)]).

This section starts from the observation that a category of presheaves is a many- sorted variety. From Theorem [3.5](#_bookmark10) we know that a large class of functors on presheaf categories have a presentation. To illustrate an application of Theorem [3.4](#_bookmark8) we give an algebraic structure of the presheaf of *λ*-terms up to *α*-equivalence. Canonical representatives for *λ*-terms up to *α*-equivalence can be obtained in different ways, for example, using the method of De Bruijn levels or the method of De Bruijn indices. Using the method of De Bruijn levels, normal forms up to *α*-equivalence are obtained by specifying well-formedness rules for *λ*-terms within a context:

1 *≤ i ≤ n x*1*,..., xn ▶ xi*

*, x*1*,..., xn, xn*+1 *▶ t*

*x*1*,..., xn ▶ λxn*+1*.t*

*, x*1*,..., xn ▶ t*1 *x*1*,..., xn ▶ t*2 *x*1*,..., xn ▶ t*1*t*2

(15)

The appropriate notion to encompass contexts and the operations allowed on them is the full subcategory F of Set with objects *n* = *{*1*,..., n}* and 0 = *∅*. The equivalence classes of *λ*-terms over a countable set of variables *V* = *{x*1*, x*2*,... }* form a presheaf in SetF, which we will denote by Λ*Vα*. Explicitly Λ*Vα*(*n*) is defined as the set of equivalence classes of *λ*-terms with the free variables contained in the set *{x*1*,..., xn}*. For any morphism *ρ* : *n → m*, Λ*Vα*(*ρ*) acts on an equivalence class of a term by substituting the free variables *xi* with *xρ*(*i*). More generally we can work with an arbitrary presheaf of variables *V* and again we can see that the *λ*-terms over *V* form a presheaf in SetF. Contexts, which correspond to natural numbers,

stratify *λ*-terms up to *α*-equivalence, and we can capture this by regarding them as the set of sorts. As we have seen in Section [2](#_bookmark1), *A* = SetF is a many-sorted unary variety, the sorts being the set of objects of F, which is isomorphic to the set of non-negative integers N. For this many-sorted variety we denote by *U* : *A →* SetN the forgetful functor and by *F* : SetN *→A* its left adjoint.

We endow F with the coproduct structure:

1

*new*

J

*n*  *i*  *n* +1

(16)

where *i* is the inclusion and *new*(1) = *n* + 1. The type constructor for context extension can be defined as a functor *δ* : *A→A* given by *δ*(*A*)(*ρ*)= *A*(*ρ* + *id*1) for all *A ∈A* and for all morphisms *ρ* in F . Let *L* : *A→A* be the functor given by

*LX* = *δX* + *X × X* (17)

If *V* is a presheaf (of variables), then an immediate consequence of Theorem 2.1 of [[11](#_bookmark33)] states that Λ*Vα* is the the free *L*-algebra over *V* . We obtain the algebraic structure of Λ*Vα* by giving an equational presentation for Alg(*L*), arising from a finitary presentation of the functor *L* and an equational presentation of the variety *A*.

In order to obtain **the equational presentation for** *A* we consider the signature:

Σ*A* = *{σ*(*i*)*|* 1 *< n,* 1 *≤ i < n}∪ {wn | n ≥* 0*}∪ {cn | n >* 0*}* (18)

*n*

with the intended interpretation being the following: *σ*(*i*) can be interpreted as the transposition (*i, i* + 1) of the set *n*, *cn* as a contraction *cn* : *n* +1 *→ n* defined by *cn*(*i*)= *i* for *i ≤ n* and *cn*(*n* + 1) = *n*, and *wn* as the inclusion of *n* into *n* + 1.

*n*

Firstly, we consider the equations coming from the presentation of the symmetric group, see for example [[24](#_bookmark46)]:

(*σ*(*i*))2(*x*)= *idn*

*n*

(*x*) 1 *≤ i < n*

*σ*(*i*)*σ*(*j*)(*x*)= *σ*(*j*)*σ*(*i*)(*x*) *j /*= *i ±* 1; 1 *≤ i, j < n* (E )

*n n n n* 1

(*σ*(*i*)*σ*(*i*+1))3(*x*)= *id* (*x*) 1 *≤ i < n −* 1

*n n n*

where *x* is a variable of sort *n*. Each permutation of the set *n* can be written as

a composition of transpositions *σ*(*i*) and we choose for each permutation such a

*n*

representation. In the next equations the permutations that will appear should be

regarded as abbreviations of their representation in terms of *σ*(*i*).

*n*

Secondly, we consider the equations coming from the presentation of the monoid of functions from *n* to *n*. [[5](#_bookmark26)] gives a presentation of this monoid for *n ≥* 4 in terms of generators of the symmetric group and an additional generator:

⎛ 1 2 3 *... n* ⎞

*A* =

⎝ 1 1 3

⎠

*... n*

Apart from the equations giving the presentation of the symmetric group, Aizenˇstat uses the following seven equations:

*Aσ*(1) = *σ*(3)*Aσ*(3) = (3*,* 4*,..., n*)*A*(3*,* 4*,..., n*) = [(1*, n*)*A*]2 = *A*

*n n n*

[*σ*(2)*A*]2 = *Aσ*(2)*A* = [*Aσ*(2)]2

*n n n*

[*σ*(2)(1*, n*)*A*]2 = [*Aσ*(2)(1*, n*)]2

*n n*

If we replace *A* in the equations above by (1*,n −* 1)(2*, n*)*wn−*1*cn−*1(1*,n −* 1)(2*, n*)

and each permutation by its representation in terms of *σ*(*i*) we will obtain the

*n*

equations which we will label by (E2). For each function from *n* to *n* we can choose

a canonical representation in terms of *σ*(*i*) and *w*

*n*

*n−*1

*cn−*1

, and one can prove that

any other representation can be reduced to this canonical one using (E1) and (E2).

Thirdly, we use the next set of equations:

*cnσ*(*n*) (*y*)= *cn*(*y*) (E3)

*n*+1

*cnwn*(*x*)= *idn*(*x*) (E4)

*σ*(*i*)

*w* (*x*)= *w*

*σ*(*i*)(*x*) 1 *≤ i < n* (E )

*n*+1 *n n n* 5

*σ*(*n*+1)*wn*+1*wn*(*x*)= *wn*+1*wn*(*x*) (E6)

*n*+2

*σ*(*i*)*cn*

*n*

(*y*) = *cn*

(*i*)

*n*+1

*σ*

(*y*) *i < n −* 1 (E7)

(*n−*1) (*n*)

*cnσ σ wn*

*n*+1 *n*+1

(*x*) = *σ*(*n−*1)*w*

*cn−*1(*x*) (E8)

*c*1*c*2*σ*(1) = *c*1*c*2 (E9)

3

*n n−*1

where *x* is a variable of sort *n* and *y* is a variable of sort *n* + 1.

We can see that the equation of (E ) corresponding to *Aσ*(1) = *A* can be obtained

2 *n*

from (E3), so we can remove it.

For each positive integers *n, k* such that *n > k*, we call a *k*-partition of *n* a *k*-uple *p* = (*i*1*,..., ik*) such that *i*1 + *···* + *ik* = *n* and 1 *≤ i*1 *≤ · · ·≤ ik*. For any *k*-partition

*p* of *n* we denote by *Cp* : *n → k* the morphism which maps the first *i*1 elements of

*n,k*

*n* to 1, the next *i*2 elements to 2 and so on, the last *ik* elements to *k*. If *n ≥* 4, if we

compose *Cp* with the inclusion of *k* in *n* we obtain a function from *n* to *n*, which

*n,k*

*n,k*

has a canonical representation in terms of *σ*(*i*) and *w*

*n*

*n−*1

*cn−*1

, abbreviated by *Np* .

Note that *Cp* = *ck ... cn−*1*Np* .

*n,k*

*n,k*

**Proposition 4.1** *Let EA denote the set of equations of the form* (*E*1) *−* (*E*9)*. Then*

*A is isomorphic to* Alg(Σ*A, EA*)*.*

**Proof (Sketch)** An exhaustive presentation of *A* can be obtained if we take an operation symbol for each morphism in F and if we consider all the equations induced by the composition of morphisms. Any morphisms of F can be generated using only

morphisms of the form *σ*(*i*)*,c ,w* . In fact each morphism *f* : *n → m* determines a

*n n n*

*k*-partition of *n* denoted by *pf* , where *k* is the cardinal of the image of *f* . If *n ≥* 4 then there exist permutations *πn* and *πm*, such that:

*f* = *πmwm−*1 *... wkck*

*... cn−*1*Npf πn*

If *n ≤* 3 then there exists permutations *πn* and *πm*, such that:

*n,k*

*f* = *πmwm−*1 *... wkck ... cn−*1*πn*

To make the notation easier we can just make the convention to denote *Npf*

*n,k*

= *idn*

for *n ≤* 3. We prove that any other representation of *f* in terms of the generators can be reduced to this canonical form using the equations (E1) *−* (E9). We will use implicitly the equations (E1) when we manipulate the permutations. Let us consider another representation of *f* . The first step is to prove that we can transform it into

a representation of the form *π' wm−*1 *... wkg* where *g* is written only in terms of

*m*

transpositions and contractions and *π'* is a permutation. This can be done using

*m*

equations (E3) *−* (E8). Next we show that there exists a permutation *τ* such that we have the following equalities in F:

*' wm−*1 *... wkτ* = *πmwm−*1 *... wk*

*π*

*m*

*τ−*1*g* = *ck*

*... cn−*1*Npf πn*

Therefore it is enough to show that we can derive the above two relations from the equations (E1) *−* (E9). The first can be obtained using (E5)*,* (E6). In order to prove the second, if *n ≥* 4, we can use (E4) and obtain:

*n,k*

*τ−*1*gπ−*1 = *ck ... cn−*1*wn−*1 *... wkτ−*1*gπ−*1

*n n*

Now *wn−*1 *... wkτ−*1*gπ−*1 can be reduced, via equations (E5)*,* (E8), to an expression

*n*

in terms of *σ*(*i*) and *w*

*n*

*n−*1

*cn−*1

. From Aizenˇstat’s result it follows that we can further

reduce this expression to *Npf* using (E1) and (E2). If *n ≤* 3 then we can just use

*n,k*

(E7) or (E9).

**A presentation for** *L* can be obtained considering for each *n ∈* N the operation symbols *lamn, appn* which semantically correspond to *λ*-abstraction and application.

The respective signature functor Σ*L* : SetN *→* SetN is given by

(Σ*LX*)*m* = *{lamm*+1*}× Xm*+1 + *{appm}× Xm × Xm* (19)

For any presheaf *V ∈A* let *ρV* : Σ*UV → ULV* be the map defined by (*lamn*+1*, t*) *'→ t ∀t ∈ V* (*n* + 1)= (*δV* )(*n*)

(*appn, t*1*, t*2) *'→* (*t*1*, t*2) *∀t*1*, t*2 *∈ V* (*n*)

The equations *EL* should correspond to the kernel pair of the adjoint transpose

: *F* Σ*UV → LV* . We will consider the set of equations of the following form:

*ρ*

*V*

*σ*(*i*)(*lam*

*, t*)= (*lam*

*, σ*(*i*)

*t*) [*t*]

*n n*+1

*n*+1

*n*+1

*wn*(*lamn*+1*, t*)= (*lamn*+2*, σ*(*n*+1)*wn*+1*t*) [*t*]

*n*+2

*cn*(*lamn*+2*, t'*)= (*lamn*+1*, σ*(*n*) *cn*+1*σ*(*n*) *σ*(*n*+1)*t'*) [*t'*]

*n*+1

*n*+2

*n*+2

(20)

*σ*(*i*)(*app ,t ,t* )= (*app , σ*(*i*)*t , σ*(*i*)*t* ) [*t ,t* ]

*n n* 1 2 *n n* 1 *n* 2 1 2

*wn*(*appn, t*1*, t*2)= (*appn, wnt*1*, wnt*2) [*t*1*, t*2]

*cn*(*appn, t*1*, t*2)= (*appn, cnt*1*, cnt*2) [*t*1*, t*2]

where *t* is a variable of sort *n* + 1, *t'* is a variable of sort *n* + 2 and *t*1*, t*2 are variables of sort *n* and *n* is an arbitrary positive integer.

**Proposition 4.2** *L is presented by ⟨*Σ*L, EL⟩.*

**Proof.** Suppose *A ∈ A*. We have to check that *LA* is a coequalizer as in dia- gram ([3](#_bookmark6)). Observe that *ρ* (*f* (*lamn*+1*, t*)) = *δ*(*A*)(*f* )(*t*) = *A*(*f* + *id*1)(*t*). Then the equations ([20](#_bookmark15)) are satisfied by *LA* because we have:

*A*

(*i*)

*σ*

*n*+1

= *σ*(*i*) + *id*1

*σ*(*n*+1)*wn*+1 = *wn* + *id*1 *σ*(*n*) *cn*+1*σ*(*n*) *σ*(*n*+1) = *cn* + *id*1

*n*

*n*+2

*n*+1

*n*+2

*n*+2

Conversely, suppose that (*f* (*lamn*+1*, t*)*,f'*(*lamn'*+1*, t'*)) is in the kernel pair of *ρ* . We have to show that *f* (*lamn*+1*, t*) and *f'*(*lamn'*+1*, t'*) can be identified in *F* Σ*UA* by different valuations of the equations. The proof follows from the fact that *A*(*f* + *id*1)(*t*)= *A*(*f'* + *id*1)(*t'*) and that for any morphisms *ρ, ρ'* which can be composed in F, we have that *ρ ◦ ρ'* + *id*1 = (*ρ* + *id*1) *◦* (*ρ'* + *id*1). Hence *f* (*lamn*+1*, t*) and (*lamm*+1*, A*(*f* + *id*1)(*t*)), can be identified in *F* Σ*UA* for any morphism *f* : *n → m*.

*A*

**Remark 4.3** The presentation of *L* depends on the operations Σ*A* used to describe

*A* = SetF but is independent of the equations *EA*.

**Representing different implementations of** *λ***-terms** If *V* is the presheaf defined by *V* (*ρ*) = *ρ* for all morphisms *ρ* in F, the free *L*-algebra over *V* gives an implementation of *λ−*terms by the De Bruijn levels method. In [[11](#_bookmark33)] it is suggested that different implementations of *λ*-terms can be obtained by equipping F with

different coproduct structures. But this implies working with a different functor than *L*. Instead, we can use another approach, namely to consider the free *L*- algebra over different presheaves of variables. For example, if *W* is the presheaf of variables defined explicitly by

*W* (*n*)= *n W* (*cn*)(1) = 1 *W* (*cn*)(*i*)= *i −* 1; *i >* 1

*W* (*wn*)(*i*)= *i* +1

*W* (*σ*(*i*))= *σ*(*n−i*)

*n n*

we obtain the presheaf Λ*Wα* of *λ−*terms implemented by the De Bruijn indices method.

# Modular coalgebraic logic

In this section we are interested in logics for (*T*2 *◦ T*1)-coalgebras. We assume that we have logics for the *Ti* described by functors *Li* with presentations *⟨*Σ*i, Ei⟩*. We know that *L* = *L*2 *◦ L*1 gives a logic for coalgebras for the functor *T* = *T*2 *◦ T*1. We also know that properties like expressiveness and completeness extend from the components *Li* to the composition *L*. Here we are interested to show how to combine the syntax and the proof systems of the two logics. In our framework, this amounts to showing how to obtain a presentation for the functor *L* = *L*2 *◦ L*1 from presentations *⟨*Σ2*, E*2*⟩* and *⟨*Σ1*, E*1*⟩*.

We know that such a presentation exists, because sifted-colimits preserving functors are closed under composition. But this in itself does not give us a recipe to compute a presentation *⟨*Σ*, E⟩* from the presentations *⟨*Σ*i, Ei⟩ in a simple modular way*. For example, in the case that *Li* : BA *→* BA, even if the Σ*i* contain only one unary operation symbol *i*, one may need an infinite set of operation symbols of arbitrary (finite) arities to present *L* = *L*2 *◦ L*1 : BA *→* BA. The reason is that operation symbols for *L* are of the form2*φ* where *φ* can be *any Boolean combination* of terms of the kind1*ψ*, or, more formally, in the notation of Section [3](#_bookmark3), operation symbols for *L* are terms in *G*1*UFG*2*UFV* .

The solution is to replace *L* by a two-sorted functor *L*¯

: BA*S →* BA*S* where we

write *S* = *{*s*,* i*}*. The intuition here is that a coalgebra *X → T*2 *◦ T*1(*X*) goes first to an intermediate state in *T*1(*X*) and then to a (proper) state in *T*2(*T*1(*X*)). This point of view introduces a 2-sorted semantics: (proper) states, of sort s, and intermediate states, of sort i. We could make this explicit using a two sorted functor *T*¯ : Set*S →* Set*S*, but we do not need to do this here. On the other hand, on the dual

side, to construct the logics, introducing a new sort for intermediate states allows us to compose presentations in a modular way. This has been used (implicitly) in our work on *π*-calculus [[7](#_bookmark30)] and goes back in coalgebraic logic to R¨oßiger [[25](#_bookmark47)] (see also Jacobs [[14](#_bookmark36)] and Cˆırstea and Pattinson [[9](#_bookmark31)]). But the technique of reflecting the structure of the type constructors in the syntax of the logic appears already in Abramsky’s use of a ‘meta-language’ in [[1](#_bookmark24)]. In our framework, the details can be

formulated as follows. First, it is convenient to introduce the notion of a two-sorted composition.

Below we consider the more general case *L*1 : *As → Ai* and *L*2 : *Ai → As*, which allows us to also treat in a modular way binary functors such as product and co- product.

**Definition 5.1** [two-sorted composition of functors] Given two functors *L*1 : *As → Ai* and *L*2 : *Ai → As* between any two categories, the two-sorted composition

of *L*1 with *L*2 is the functor *L*¯ : *Ai × As → Ai × As* mapping *A* = (*A*i*, A*s) to

(*L*¯*A*)s = *L*2(*A*i) and (*L*¯*A*)i = *L*1*A*s.

This composition is symmetric: Swapping *L*1 and *L*2 just means that the indices i and s change role. It is therefore tempting to suppress the distinction between 1 and i and between 2 and s in our notation. We do not do this because we want to use the notation (*−*)1 to refer to the functor *L*1 and the notation (*−*)i to refer to a projection onto sort i.

The next proposition ensures that we can extract the initial *L*2 *◦ L*1-algebra from the initial algebra of the two-sorted composition. (We continue to write again *L*2*L*1 instead of *L*2 *◦ L*1).

**Proposition 5.2** *Consider categories A*i*, A*s *which are lfp and two ﬁnitary functors L*1 : *As → Ai and L*2 : *Ai → As. Let L*¯ *be the two-sorted composition of L*1 *with L*2*. Then the* s*-component of the initial L*¯*-algebra is the initial L*2*L*1*-algebra.*

**Proof.** *Ai × A*s is lfp and *L*¯ is finitary. Therefore, the initial *L*¯-algebra is the

colimit of the initial algebra chain *L*¯*n*0 where 0 denotes the initial object and *n* runs through finite ordinals. As colimits are calculated sort-wise, it is enough to show that the projected sequence (*L*¯*n*0)s has the same colimit as the initial sequence of *L*2*L*1, which is easy to see as the latter sequence is a subsequence of the former.

As the relation between logics and coalgebras is provided by the initial algebras, the proposition tells us that we do not loose anything if we present *L*¯ instead of *L*2*L*1. It is obvious how to do this.

**Theorem 5.3** *Consider (many-sorted) varieties A*i *× A*s *and two functors L*1 :

*As → Ai and L*2 : *Ai → As with presentations ⟨*Σ1*, E*1*⟩ and ⟨*Σ2*, E*2*⟩, respectively.*

*Then ⟨*Σ¯ *, E*¯*⟩ is a presentation of the two-sorted composition L*¯ *of L*1 *with L*2 *as follows.*

(Σ¯ *X*)s = Σ2*X*i (Σ¯ *X*)i = Σ1*X*s

*where we use that the signatures* Σ1*,* Σ2 *are given by functors* Set*S*2 *→* Set*S*1 *,* Set*S*1 *→* Set*S*2 *and X* = (*X*i*, X*s) *denotes and element of* Set*S*1 *×* Set*S*2 *. Equations are given by E*¯s = *E*2*, E*¯i = *E*1*.*

**Example 5.4** Let us illustrate this theorem using more familiar notation. To be specific, we assume that *Ai* and *As* are both BA. We write *▶*i *ψ* and *▶*s *φ* to assert that *ψ, φ* are formulas of sort i*,* s, respectively. The theorem then states that formulas of both sorts are closed under Boolean operations and, for all *n*-ary operation symbols *σi* in Σ*i*, formulas are closed under

*▶*i *ψ*1*,..., ▶*i *ψn*

*▶*s *σ*2(*ψ*1*,..., ψn*)

*▶*s *φ*1*,..., ▶*s *φn*

*▶*i *σ*1(*φ*1*,..., φn*)

The axioms are given by equations *E*1*, E*2, sortwise. The rules of the calculus are those of equational logic. The only rules that make the two sorts interact are the congruence rules:

*▶*i *ψ*1 = *ψ' ,..., ▶*i *ψn* = *ψ' ▶*s *φ*1 = *φ' ,..., ▶*s *φn* = *φ'*

1 2 1 *n*

*▶*s *σ*2(*ψ*1*,..., ψn*)= *σ*2(*ψ' ,..., ψ'* ) *▶*i *σ*1(*φ*1*,..., φn*)= *σ*1(*φ' ,..., φ'* )

1 *n* 1 *n*

Here, we use *▶*i *ψ* = *ψ'* and *▶*s *φ* = *φ'* to denote derivability of equations of the respective sorts.

**Example 5.5** In the literature, one often considers inductively defined classes of functors [[25](#_bookmark47),[14](#_bookmark36),[9](#_bookmark31)], such as, for example, the class of so-called Kripke polynomial functors *T* : Set *→* Set built according to

*T* ::= *Id | KC | T* + *T | T × T | T ◦ T |У*

where *Id* is the identity functor, *KC* is the constant functor that maps all sets to a finite set *C*, *У* is covariant powerset.

Typically, these functors are not only built from basic ingredients (such as *Id, KC, У*) and composition, but also from binary operations (such as +*, ×*). To give an exam- ple, the functor *L*+ : BA *×* BA *→* BA capturing the logic of + : Set *×* Set *→* Set has the following presentation.

There are two unary operation symbols [*κ*1] and [*κ*2]. Equations specify that the [*κi*] preserve finite joins and binary meets and that [*κ*1]*a*1*Λ*[*κ*2]*a*2 = *⊥*, [*κ*1]*TV*[*κ*2]*T* = *T*, *ч*[*κ*1]*a*1 = [*κ*2]*TV* [*κ*1]*чa*1, *ч*[*κ*2]*a*2 = [*κ*1]*TV* [*κ*2]*чa*2.

If *L* is now the logic of *T* , then the logic of *T* + *T* is given by the presentation of

the two-sorted composition of BA *⟨L,L⟩* BA *×* BA *L*+

*—→*

*—→*

BA according to Theorem [5.3](#_bookmark17).

# Uniform completeness proofs

In this section we show how to associate to an arbitrary set-functor *T* a functor *L* on BA and a semantics *δ* : *LP → PT* so that the resulting logic is complete. The definition of *L* from *T* is the same as in [[20](#_bookmark44),[19](#_bookmark41)], but as we do not insist on strong completeness [2](#_bookmark19) here, *we don’t need to put any assumptions on T* . Instead we use

2 A logic is strongly complete if, whenever *φ* holds in all models satisfying a possibly infinite set of formulas Γ, then one can also derive *φ* from Γ. Strong completeness is closely related to compactness. So, for example, the procedure below will not give rise to strongly complete logics if *T* is the probability distribution functor or if *TX* = *A × X* for an infinite set *A*.

an induction along the final sequence as first done in Pattinson [[23](#_bookmark48)] and adapted to the setting of functorial logics over BA in [[17](#_bookmark39)].

**Definition of *L*.** First, let us recall from [[20](#_bookmark44),[19](#_bookmark41)] the definition of *L* from *T* (see also Klin [[16](#_bookmark38)]). The essential ingredients are as follows. Two contravariant functors *P* and *S* that are adjoint on the right

z\_*. P*

\_*X*  *s*

*T*

*L A*,*,* (21)

*S*

*I*

*A*0

where *A* is lfp with a small subcategory *A*0 of finitely presentable objects. We then define *L* on *A*0, eliding the inclusion *I*, as

*LA* = *PTSA*

and extend *L* continuously from *A*0 to *A*. Note that *L* thus defined preserves filtered colimits, whereas *PTS* need not to do so.

**Example 6.1** Take *A* = BA and *X* = Set. Then *P* is contravariant powerset and *S* takes ultrafilters. On arrows, *P* and *S* map a function to its inverse image. The adjunction restricts to a dual equivalence between finite Boolean algebras and finite sets. The ultrafilters of a finite Boolean algebra *A* are the atoms of *A*, that is, those elements *a ∈ A* such that there are no elements strictly between bottom and *a*. Thus, on finite Boolean algebras, the duality reduces to the well known fact that every finite Boolean algebra is isomorphic to the powerset of its atoms. We will also make use of the fact that the finitely presentable Boolean algebras coincide with the finite ones.

**Definition of *δ* : *LP → PT* .** The idea that the semantics of a logic for coalgebras should be described by a natural transformation *LP → PT* goes back to [[17](#_bookmark39),[6](#_bookmark28)]. The following definition is again from [[20](#_bookmark44)].

*P X*,*,*

*LP*,*X, δX* *P T*,*X,* (22)

*ci Lci PTc*

*i*

*A LA*  *∼*= *P TSA*

*i* *i*

*i*

*PX* is a filtered colimit *ci* : *Ai → PX*. Under the adjunction, this cocone corre- sponds to a cone *c* : *X → SAi* which is turned into a cocone under *PT* (recall that *P* is contravariant). Now *δX* exists uniquely, since *L* preserves filtered colimits .

*i*

Intuitively, the logic *L* is complete if any two formulas identified in the semantics, are already identified in the syntax. This is the content of the following lemma.

**Assumption:** From now on we take *A* = BA*S'* and *X* = Set*S'* with the functors

*P* and *S* sort-wise as in Example [6.1](#_bookmark20).

**Lemma 6.2** *δX as deﬁned above is injective.*

**Proof.** Consider two distinct *φ*1*, φ*2 *∈ LPX*. By filteredness, we find some *Ai* and

*φ' ∈ Ai* such that *ci*(*φ'* )= *φj*. Moreover, since in BA the finitely presentable objects

*j j*

are closed under quotients, we can assume *ci* to be injective. The following fact is easily proved.

**Claim:** Let *A* be finite. *c* : *A → PX* is injective iff the adjoint transpose *c* : *X →*

*SA* is surjective.

Indeed, by the laws of adjunction and *A* being finite, we have that *c* is *A ∼*= *PSA Pc PX*; now *Pc* = (*c*)*−*1 is injective iff *c* is surjective, which proves the claim. Using that *T* , as any functor on Set*S'* , preserves surjective maps and that *P* maps surjective maps to injective BA*S'* -homomorphisms, we conclude that *PT c* is injective, hence *δX* (*φ*1) */*= *δX* (*φ*2).

*i*

*—→*

**Theorem 6.3** *The logic given by L as deﬁned above is complete for T-coalgebras.*

**Proof.** (The proof is essentially the one from [[17](#_bookmark39)], where the reader can find the missing technical details.) Let *L* : BA*S' →* BA*S'* be the functor defined above. *L* preserves filtered colimits and therefore, using a special property of BA and following [[20](#_bookmark44), Proposition 3.4], *L* preserves sifted colimits. It follows that *L* has a presentation, which induces an equational logic, which in turn can be written in the usual modal- logic style, using the correspondences between equations *φ* = *ψ* and formulas *φ ↔ ψ* and between formulas *φ* and equations *φ* = *T*.

The semantics of an *L*-formula wrt a coalgebra *ξ* : *X → TX* is determined by the arrow **[***—*]](*X,ξ*) from the initial *L*-algebra to the algebra *LPX → PTX → PX*. Because of the naturality of *δ*, the semantics wrt to all coalgebras is determined by the semantics wrt to the final coalgebra. Since we don’t assume that the final coalgebra exists, we replace it by the corresponding final sequence *Tn * which is

defined as follows. We denote by  = *T* 0  the final object in Set*S'* . *p*

0

: *T * *→ *

is given by finality and *pn*+1 : *T* (*Tn * ) *→ Tn * is defined to be *Tpn*. We think of the *Tn * as approximating the final coalgebra. [3](#_bookmark21) In the same way as any coalgebra *ξ* : *X → TX* has a unique arrow into the final coalgebra, there are canonical arrows *ξn* : *X → Tn * to the approximants of the final coalgebra, defined inductively by *ξn*+1 = *T* (*ξn*) *◦ ξ*. The idea now is to interpret a formula *φ* ‘of depth *n*’ as a subset

[[*φ*]]*n* of *Tn * . The semantics of *φ* in *X* is then *ξ−*1([[*φ*]]*n*). [4](#_bookmark22) To say what it means for

*n*

a formula to be of depth *n* we need the initial sequence of *L*, which we define next.

Since *L* is finitary the initial algebra is the colimit of the sequence *Ln * defined as

follows. We denote by  = *L*0 the initial object in BA*S'* . *e*

0

:  *→ L* is given by

initiality and *en*+1 : *Ln * *→ L*(*Ln * ) is defined to be *Len*. Since *L* preserves sifted colimits and hence injective maps [[20](#_bookmark44), Corollary 4.10], all maps in the sequence are injective. This means that we can consider the initial *L*-algebra as a union of its approximants *Ln * . We call the elements of *Ln * formulas of depth *n*. The semantics

3 Indeed, if we let run the final sequence through all ordinals, we obtain the final coalgebra as a limit if it exists, see Adamek and Koubek [[2](#_bookmark25)].

4 This point of view has been elaborated in [[18](#_bookmark40)].

of a formula of depth *n* is given by a BA*S'* -morphism **[***—*]] : *Ln * *→ PTn * as follows.

*n*

*PTn*  *P pn* *P Tn*+1  *...*



*P*,*,*

[[*−*]]0

*Pp*0 *. . .*

*e*0

*. . .*

(23)

,*,*

[[*−*]]*n*

*Ln*



*en*

,*,*

[[*−*]]*n*+1

*Ln* +1 

*...*

[[*—*]]0 is given by initiality (and is actually the identity). **[***—*]]*n*+1 is defined to be *δT n * *◦ L*([[*—*]]*n*). Observe that the semantics of a formula is independent of the particular approximant we choose (all squares in the diagram commute). Moreover, given a coalgebra *ξ* : *X → TX* and a formula of depth *n*, the semantics via the initial

*n*

*L*-algebra and the semantics via the final sequence coincide: **[***φ*]]

(*X,ξ*)

= *ξ−*1([[*φ*]]

*n*).

Since *δ* is injective and *L* preserves injective maps, all **[***—*]]*n*, *n ∈* N, are injective.

To show completeness, suppose *φ*1 */*= *φ*2 in the initial *L*-algebra. We find an ap- proximant *Ln * , in which *φ*1 and *φ*2 are different. Any one-sided inverse *i* of *p*0 gives rise to a *T* -coalgebra *ξ* = *Tn*(*i*) with carrier *Tn * . We have **[***φ*]](*T n * *,ξ*) = [[*φ*]]*n*. Now injectivity of **[***—*]]*n* shows that (*Tn * *, ξ*) provides a counter-example for *φ*1 = *φ*2.

# References

1. S. Abramsky. Domain theory in logical form. *Ann. Pure Appl. Logic*, 51, 1991.
2. J. Ad´amek and V. Koubek. On the greatest fixed point of a set functor. *Theoret. Comput. Sci.*, 150, 1995.
3. J. Ad´amek and J. Rosicky´. *Locally Presentable and Accessible Categories*. CUP, 1994.
4. J. Ad´amek and J. Rosicky´. On sifted colimits and generalized varieties. *Th. Appl. Categ.*, 8, 2001.
5. A. Y. Aizenˇstat. Defining relations of finite symmetric semigroups. *Math. Sb. N. S.*, 45, 1958.
6. M. Bonsangue and A. Kurz. Duality for logics of transition systems. In *FoSSaCS’05*.
7. M. Bonsangue and A. Kurz. Pi-calculus in logical form. In *LICS’07*.
8. M. Bonsangue and A. Kurz. Presenting functors by operations and equations. In *FoSSaCS’06*.
9. C. Cˆırstea and D. Pattinson. Modular construction of modal logics. In *CONCUR’04*.
10. M. Fiore and C.-K. Hur. Equational systems and free constructions. In *ICALP’07*.
11. M. Fiore, G. Plotkin, and D. Turi. Abstract syntax and variable binding. In *LICS’99*.
12. M. Gabbay and A. Pitts. A new approach to abstract syntax involving binders. In *LICS’99*.
13. M. Hofmann. Semantical analysis of higher-order abstract syntax. In *LICS’99*.
14. B. Jacobs. Many-sorted coalgebraic modal logic: a model-theoretic study. *Theor. Inform. Appl.*, 35, 2001.
15. B. Klin. The least fibred lifting and the expressivity of coalgebraic modal logic. In *CALCO’05*.
16. B. Klin. Coalgebraic modal logic beyond sets. In *MFPS’07*, 2007.
17. C. Kupke, A. Kurz, and D. Pattinson. Algebraic semantics for coalgebraic logics. In *CMCS’04*.
18. A. Kurz and D. Pattinson. Coalgebraic modal logic of finite rank. *Math. Structures Comput. Sci.*, 15, 2005.
19. A. Kurz and J. Rosicky´. The Goldblatt-Thomason-theorem for coalgebras. In *CALCO’07*.
20. A. Kurz and J. Rosicky´. Strongly complete logics for coalgebras. July 2006. Submitted.
21. L. Moss. Coalgebraic logic. *Ann. Pure Appl. Logic*, 96, 1999.
22. D. Pattinson. Semantical principles in the modal logic of coalgebras. In *STACS’01*.
23. D. Pattinson. Coalgebraic modal logic: Soundness, completeness and decidability of local consequence.

*Theoret. Comput. Sci.*, 309, 2003.

1. D. Robinson. *A Course in the Theory of Groups*. Springer, 1996.
2. M. R¨oßiger. Coalgebras and modal logic. In *CMCS’00*.
3. L. Schr¨oder. Expressivity of Coalgebraic Modal Logic: The Limits and Beyond. In *FoSSaCS’05*.