Towards a Duality Result in the Modal Logic of Coalgebras

Bart Jacobs

Dep. Comp. Sci., Univ. Nijmegen,

P.O. Box 9010, 6500 GL Nijmegen,

Email: `bart@cs.kun.nl` URL: `http://www.cs.kun.nl/~bart`

January 3, 2000

Abstract

This paper forms a step in the development of the recently emerged connection between coalgebra and modal logic. It introduces (back-and-forth) transformations between coalgebras of simple polynomial functors and certain Boolean algebras with operators (BAOs). Categorically, these transformations take the form of an adjunction. The BAO associated with a coalgebra can be used for specification, *e.g.* of classes in object-oriented languages.

1 Introduction

Coalgebras are simple mathematical structures that can be seen as very general state-based dynamical systems. Examples include automata and transition systems, but also programs (as state transformers) and classes in object-oriented languages, see [JR97, Rut99, Rei95]. Modal logic is a logic for dynamical systems. It is therefore no surprise that there are connections between the areas of coalgebra and modal logic [Mos99, Röß00, Röß99, Kur00, Jac99]. This papers takes a step in the development of this connection. It takes the classical (back-and-forth) transformations between Kripke structures (transition models of modal logic) and Boolean algebras with operators (BAOs) as starting point, see [Lem66, Gol93, BdRV], and establishes a similar result for certain coalgebras instead of Kripke structures. Thus it establishes a relation (in the form of an adjunction) between coalgebras and BAOs. This construction raises many further questions, most of which are left unanswered at this stage.

Coalgebras have interfaces determining the access points for the outside world. These interfaces are certain functors $T\colon \mathbf{Sets} \to \mathbf{Sets}$ on the category **Sets** of sets and functions—since we restrict ourselves to set-based coalgebras. Here we use so-called polynomial functors, which are built-up from the identity and constant functors using products, coproducts and exponents (with a constant). These functors suffice for describing many interfaces of classes in object-oriented languages, see [Jac96, Jac00]. For a functor T there is a category $\mathbf{CoAlg}(T)$ of coalgebras $X \to T(X)$ with interface T. This paper introduces a category $\mathbf{BAO}(T)$ of Boolean algebras with operators of (polynomial) type T. The main result (see Theorem 8.4) is then an adjunction of the form:

$$\begin{tikzcd} \mathbf{BAO}(T) \arrow[d, bend right=50, "\mathsf{spec}"'] \arrow[d, phantom, "\dashv"] \\ \mathbf{CoAlg}(T)^{\mathrm{op}} \arrow[u, bend right=50, "\mathcal{P}"'] \end{tikzcd} \qquad (1)$$

It describes how for a coalgebra $X \to T(X)$ the Boolean algebra $\mathcal{P}(X)$ of predicates on the state space X carries suitable modal operators (and observers). And in the reverse direction, how for such a Boolean algebra with operators B the set of ultrafilters (or points) $\mathsf{spec}\, B$ carries a coalgebra structure $\mathsf{spec}\, B \to T(\mathsf{spec}\, B)$. Doing these transformations twice yields certain canonical "unit" and "counit" maps $B \to \mathcal{P}(\mathsf{spec}\, B)$ and $X \to \mathsf{spec}\, \mathcal{P}(X)$ which are an essential ingredient of the adjunction (1).

Every adjunction $\left(\mathbb{C} \rightleftarrows \mathbb{D}\right)$ gives rise to an "adjoint" equivalence of categories [Lan71, IV, §§4], namely between the full subcategories $\mathbb{C}_{\cong}$ and $\mathbb{D}_{\cong}$ of those objects for which the unit or counit map of the adjunction is an isomorphism. In the above case (1) we could write this equivalence as a duality $\mathbf{BAO}(T)_{\cong} \simeq \mathbf{CoAlg}(T)^{\mathrm{op}}_{\cong}$. However, we have no intrinsic characterisation of the objects in $\mathbf{BAO}(T)_{\cong}$ or $\mathbf{CoAlg}(T)_{\cong}$, so the duality does not provide much insight at this stage.

The standard reference on (Stone) duality is [Joh82]. It gives a central position to the adjunction between complete Heyting algebras (or locales) and topological spaces,

$$\begin{tikzcd} \mathbf{cHA} \arrow[d, bend right=50, "\mathrm{points}"'] \arrow[d, phantom, "\dashv"] \\ \mathbf{Sp}^{\mathrm{op}} \arrow[u, bend right=50, "\mathrm{opens}"'] \end{tikzcd} \qquad (2)$$

namely as the mother of all dualities (in the book). This adjunction immediately gives a duality $\mathbf{cHA}_{\cong} \simeq \mathbf{Sp}^{\mathrm{op}}_{\cong}$ between *spatial* complete Heyting algebras and *sober* topological spaces. But it can be further refined to *e.g.* the duality between distributive lattices and coherent spaces [Joh82, II, §§3.4], and to the famous Stone duality between Boolean algebras and Stone spaces (coherent Hausdorff spaces) [Joh82, II, §§4.4]. There is a resemblance between the adjunctions (1) and (2), but it is still too early to say what the rôle of the adjunction (1) will be.

The standard duality in modal logic [Lem66, Gol93, BdRV] is between Kripke structures and certain Boolean algebras with operators. A Kripke structure consists of a set X with an (unlabeled) transition relation $R \subseteq X \times X$. Equivalently, it is a coalgebra $X \to \mathcal{P}(X)$ of the powerset functor $\mathcal{P}$. A serious restriction in the present paper is that this powerset functor is not allowed to occur in the kind of polynomial functors that we consider (in contrast to [Röß99, Jac99]). Regrettably, a unification between the adjunction (1) of this paper and the standard results from modal logic is still an open topic.

The reasons for excluding the powerset functor are somewhat technical. Without it, the modal operators induced by a coalgebra of a polynomial functor are affine functions (additionally preserving binary joins $\vee$, besides finite meets $\top, \wedge$). This property is used for the construction of coalgebras from operators (and observers), see Section 7, relying on Lemma 7.1, especially point (iii) where the property is used. It seems that coproduct $+$ and powerset $\mathcal{P}$ do not mix very well. And since our interest stems from coalgebraic specification, where the coproduct are extremely useful, we chose to exclude the powerset. How to adapt the coalgebra constructions in this paper so that powersets can be included is a topic of further research. Further, as will be shown in Section 4, the adjunction in Theorem 8.4 that we will establish does not exist for Kripke structures (coalgebras of the poserset functor) and BAOs. This is also an impediment to unification.

The starting point of this paper was the seemingly easy plan to combine the modal operators defined in [Jac99, Section 4] for coalgebras with the canonical (coalgebraic) model construction from [Röß00, Röß99]. The latter uses maximally consistent sets of formulas, which, from an algebraic perspective, are ultrafilters. This combination turned out to be much harder than expected. In particular, many subtle properties of "predicate lifting" (underlying the modal operator definitions in [Jac99]) were needed. These are collected in Sections 6 and 7 below, and form the main technical contribution of this paper. These properties allow us to decompose and assemble coalgebras. They result in suitable requirements on the operators and observers of the Boolean algebras that we use. These requirements are the (algebraic) counterparts of the axioms used in [Röß00, Röß99]. Once this is worked out, the resulting adjunction (1) between coalgebras and BAOs follows easily.

We impose the restriction on our polynomial functors that the constants they contain should be (non-empty) *finite* sets. This is needed because we work with not-necessarily complete Boolean algebras, as is standard in modal logic. Lifting the finiteness restriction leads to the use of complete Boolean algebras, and of completely prime filters (instead of prime filters, *i.e.* ultrafilters). The constructions that we use seem to work for this more general situation as well.

As already mentioned, this work is only a first step in this area. Many problems are left open, and applications (like to a Birkhoff result in this setting) are not considered. We do however motivate the usefulness of the BAO associated with a functor by considering the rôle that the operators and observers can play in coalgebraic specification. The example that we use serves at the same time as an introduction to the ultrafilter construction. Such modal logics may be of interest in future work on model checking for coalgebras.

Duality theory is an area where the use of elementary category theory directly pays off, in the sense that formulations become more compact and understandable. For this paper it suffices to know about functors, natural transformations and adjunctions. These notions occur in any textbook on category theory, such as [Wal91, BW90, Lan71, Bor94].

This paper is organised as follows. It starts with three preliminary sections on ultrafilters, on polynomial functors, and on Kripke structures and BAOs. Section 5 introduces a coalgebraic specification of stacks via modal operators, and shows how to define ultrafilter extensions of stack coalgebras. The subsequent two sections 6 and 7 contain the main technical work on paths, predicate lifting, operators and observers, decomposition and assembly of coalgebras. The final Section 8 discusses how to turn coalgebras into BAOs and vice-versa, leading to the main result (Theorem 8.4) establishing an adjunction (1).

2 Order theoretic preliminaries

This section starts with some order theoretic preliminaries culminating in ultrafilters of Boolean algebras, and their rôle in Stone's duality result. Let X be a poset with finite meets $\top, \wedge$. A *filter* of X is a subset $U \subseteq X$ which is closed under meets $\top, \wedge$ and is also upwardly closed. Thus: $\top \in U$, and $x, y \in U \Rightarrow x \wedge y \in U$, and $y \geq x \in U \Rightarrow y \in U$. For an arbitrary subset $S \subseteq X$ we write $\uparrow S = \{x \in X \mid \exists y \in S. x \geq y\}$ for the least upwardly closed subset containing S. When S is a singleton $\{x\}$, we write $\uparrow x$ for $\uparrow\{x\}$, and obtain a (so-called principal) filter. More generally, the least filter containing S is the set $\Uparrow S = \uparrow\{\bigwedge \alpha \mid \alpha \subseteq S \text{ finite}\}$.

Similarly, an *ideal* of a poset X with finite joins $\perp, \vee$ is a subset which is closed under these joins $\perp, \vee$ and which is downwards closed. The downsets $\downarrow x = \{y \in X \mid y \leq x\}$ are (principal) ideals.

Let X now be a Boolean algebra (BA), *i.e.* a poset with finite meets and joins and a complement operators $\neg$. A typical example is the powerset $\mathcal{P}(A)$ of subsets (also called predicates) of an arbitrary set A. An *ultrafilter* of X (or also called a maximal or prime filter) is a filter $U \subseteq X$ which satisfies: for each $x \in X$, either $x \in U$ or $\neg x \in U$, but not both. As a consequence, $\perp \notin U$. Also, if $x \vee y \in U$, then either $x \in U$ or $y \in U$—because, if not, then $x \notin U$ and $y \notin U$ yields $\neg x \in U$ and $\neg y \in U$, and thus $\neg x \wedge \neg y = \neg(x \vee y) \in U$, contradicting $x \vee y \in U$. This can be generalised to finite disjunctions: if I is finite, and $\bigvee_{i \in I} x_i \in U$, then $x_i \in U$, for some $i \in I$. We write $\dot{\vee}$ for the exclusive disjunction $x \mathbin{\dot{\vee}} y = (x \vee y) \wedge \neg(x \wedge y)$, and $\dot{\bigvee}$ for its generalisation over finite sets, defined as $\dot{\bigvee}_{i \in I} x_i = \big(\bigvee_{i \in I} x_i\big) \wedge \big(\bigwedge_{i,j \in I}\{\neg(x_i \wedge x_j) \mid i \neq j\}\big)$. Then $\dot{\bigvee}_{i \in I} x_i \in U$ implies $x_i \in U$ for a *unique* $i \in I$. Let's write $x \supset y = \neg x \vee y$ for implication in a Boolean algebra, and $x \supset\subset y$ for bi-implication $(x \supset y) \wedge (y \supset x)$. Then $x \supset y \in U$ and $x \in U$ implies $y \in U$. Ultrafilters in a Boolean algebra are the algebraic counterparts of maximally consistent theories in logic.

In powerset Boolean algebras $\mathcal{P}(A)$ meets $\perp, \wedge$ are $A \subseteq A$ and intersection $\cap$, and joins $\perp, \vee$ are $\emptyset \subseteq A$ and union $\cup$. The exclusive disjunction $\dot{\bigvee}_{i \in I} P_i$ of subsets $P_i \subseteq A$ is $\{a \in A \mid a \in P_i \text{ for precisely one } i \in I\}$. For a function $f\colon A \to B$, the induced inverse image function $f^{-1}\colon \mathcal{P}(B) \to \mathcal{P}(A)$ given by $U \mapsto \{a \in A \mid f(a) \in U\}$ preserves all Boolean operations, including exclusive disjunctions.

The following result is often useful. Its proof depends on the Axiom of Choice (in the form of Zorn's Lemma), see [Joh82, I, 2.3 and 2.4] or [DP90, Theorem 9.13].

2.1. Lemma (Ultrafilter Lemma). *Let F be a filter and I be an ideal of a Boolean algebra X, with $F \cap I = \emptyset$. Then there is an ultrafilter $U \subseteq X$ with $F \subseteq U$ and $U \cap I = \emptyset$.* $\square$

The next result relates filters and ultrafilter. It is used for example in [Lem66] (see §§III, Lemma 2 and 3). It resembles the "Scott open filter lemma", see *e.g.* [Vic89, Lemma 8.2.2].

2.2. Lemma. *Let X be Boolean algebra with a filter $F \subseteq X$. Then*

$$F = \bigcap\{U \in \mathsf{spec}\, X \mid F \subseteq U\}.$$

Proof. The direction ($\subseteq$) is immediate. For the reverse implication ($\supseteq$) assume $\forall U \in \mathsf{spec}\, X. F \subseteq U \Rightarrow x \in U$ and $x \notin F$. We first show that $\perp$ is then not in the filter $\Uparrow(F \cup \{\neg x\})$, because it implies $x \in F$. If $\perp \geq \bigwedge \alpha$ for a finite set $\alpha \subseteq F \cup \{\neg x\}$ we can distinguish whether $\neg x$ is in α or not. If it is not, then $\perp \in F$ and thus certainly $x \in F$. If $\neg x \in \alpha$, write $\alpha = \{\neg x\} \cup \beta$ with $\beta \subseteq F$ and $\neg x \wedge \bigwedge \beta = \perp$. Then $\bigwedge \beta \leq x$, and thus $x \in F$.

Now we can apply the previous lemma to the filter $\Uparrow(F \cup \{\neg x\})$ and the ideal $\{\perp\}$. It yields an ultrafilter U with $F \cup \{\neg x\} \subseteq U$. But then $F \subseteq U$ and $x \notin U$, contradicting the assumption. □

We call a function between Boolean algebras *affine* if it preserves binary joins $\vee$. And it will be called *linear* if it is affine and preserves the bottom element $\perp$. The next lemma is trivial, but worth emphasising because it plays a crucial rôle in the sequel.

2.3. Lemma. *Let X, Y be a Boolean algebras with an affine function $f: X \to Y$ between them preserving finite meets $\top, \wedge$. If $U \subseteq Y$ is an ultrafilter with $f(\perp) \notin U$, then $f^{-1}(U) = \{x \in X \mid f(x) \in U\}$ is an ultrafilter of X.* □

We write $\mathsf{spec}\, X$ for the set of ultrafilters of a Boolean algebra X, and call it the *spectrum* of X. It is a special topological space, namely a *Stone space*. Its opens can be described as indexed by ideals, namely as $\mathcal{O}(I)$ for ideals $I \subseteq X$, where $\mathcal{O}(I) = \{U \in \mathsf{spec}\, X \mid U \cap I \neq \emptyset\}$. It is easy to see that $I \subseteq J$ implies $\mathcal{O}(I) \subseteq \mathcal{O}(J)$, but with a little more effort one sees that the converse[1] also holds: if $\mathcal{O}(I) \subseteq \mathcal{O}(J)$ and $x \in I$ but $x \notin J$, then Lemma 2.1 yields an ultrafilter U with $\uparrow x \subseteq U$ and $U \cap J = \emptyset$. Then $U \notin \mathcal{O}(J)$, and thus $U \notin \mathcal{O}(I)$. But $x \in I \cap U$.

From this equivalence $I \subseteq J \Leftrightarrow \mathcal{O}(I) \subseteq \mathcal{O}(J)$ it follows easily that the compact open sets in $\mathsf{spec}\, X$ are of the form $\eta(x) = \mathcal{O}(\downarrow x) = \{U \in \mathsf{spec}\, X \mid x \in U\}$, for $x \in X$. They form a Boolean algebra, and every open can be obtained as union of such compact opens. This makes the spectrum of X a Stone space. The map $\eta: X \to \mathcal{P}(\mathsf{spec}\, X)$ is injective, since by Lemma 2.2 $\uparrow x = \bigcap\{U \mid \uparrow x \subseteq U\} = \bigcap \eta(x)$, and thus $\eta(x) = \eta(y)$ implies $\uparrow x = \uparrow y$, and thus $x = y$. The functoriality aspect of Stone duality relies on the fact for a homorphism $f: X \to Y$ of Boolean algebras (preserving $\top, \wedge, \perp, \vee$), the inverse image function $f^{-1}: \mathcal{P}(Y) \to \mathcal{P}(X)$ sends ultrafilters to ultrafilter—*i.e.* restricts to $f^{-1}: \mathsf{spec}\, Y \to \mathsf{spec}\, X$, by the previous lemma.

For further details on Stone duality, we refer to the standard monograph [Joh82].

3 Polynomial functors

We shall be using a particular collection of functors $T: \mathbf{Sets} \to \mathbf{Sets}$, as interfaces of coalgebras. These so-called polynomial functors are built up inductively from the identity and constants, using products, coproducts and exponents (with constants). Products of sets X, Y will be written as $X \times Y$, with projection functions $X \xleftarrow{\pi_1} X \times Y \xrightarrow{\pi_2} Y$. Coproducts (or disjoint unions, or sums) are denoted by $X + Y$, where, for example $X + Y = \{(x, 0) \mid x \in X\} \cup \{(y, 1) \mid y \in Y\}$. It comes with coprojection functions $X \xrightarrow{\kappa_1} X + Y \xleftarrow{\kappa_2} Y$. The coprojections are injective and disjoint (in the sense that $\kappa_1(x) \neq \kappa_2(y)$, for all $x \in X$ and $y \in Y$). The set 1 will be a singleton set, typically written as $1 = \{*\}$.

The collection of *polynomial functors* $\mathbf{Sets} \to \mathbf{Sets}$ that we use arises in the following way.

[1]Basically this says that the locale of ideals $\mathsf{Idl}(X)$ is spatial, see [Joh82, II, Theorem 3.4].

1. The identity functor $\mathrm{Id}: \mathbf{Sets} \to \mathbf{Sets}$ is polynomial.
2. For each non-empty set D, the constant functor $D: \mathbf{Sets} \to \mathbf{Sets}$, given by $X \mapsto D$ and $(f: X \to Y) \mapsto \mathrm{id}_D$, is polynomial.
3. The product $X \mapsto T_1(X) \times T_2(X)$ of two polynomial functors T_1, T_2 is polynomial.
4. The coproduct $X \mapsto T_1(X) + T_2(X)$ of two polynomial functors T_1, T_2 is also polynomial.
5. For a polynomial functor T_1, and an arbitrary non-empty set E the exponent functor $X \mapsto T_1(X)^E$ is also polynomial.

An example of a polynomial functor is

$$X \longmapsto \mathbb{Z} \times X^A \times \Big(1 + \big(A \times X\big)\Big).$$

Often in the sequel we restrict ourselves to those polynomial functors T all of whose constants (like D in 1. and E in 5.) are finite. In that case the exponent clause 5. is not really needed, because for a finite set $E = \{e_1, \ldots, e_n\}$ there is an isomorphism

$$Y^E \;\cong\; \underbrace{Y \times \cdots \times Y}_{n \text{ times}}$$

We still include the exponent case in our definition of polynomial functors because:

- exponents are convenient in specification, see the stack example in Section 5;
- Many of the results in this paper do not depend on the finiteness assumption (for constants in polynomial functors). Hence, for a generalisation of the main result (Theorem 8.4) to complete Boolean algebras and completely prime filters it is good know precisely where extra work is needed.

A *coalgebra* of a (polynomial) functor $T: \mathbf{Sets} \to \mathbf{Sets}$ consists of a set X, usually called the state space or set of states, together with a function $c: X \to T(X)$, giving the operations of the coalgebra. Section 5 describes stacks as an example of a coalgebra.

A *(homo)morphism of coalgebras* from $X \xrightarrow{c} T(X)$ to $Y \xrightarrow{d} T(Y)$ is a function $f: X \to Y$ between the underlying state spaces which commutes with the operations: $d \circ f = T(f) \circ c$. We write $\mathbf{CoAlg}(T)$ for the resulting category of coalgebras of the functor T.

4 A brief review of Kripke structures and Boolean algebras with operators

This section sets the stage by recalling some standard results (see [Lem66, Gol93]) about the relation between dynamic models (Kripke structures) and algebraic models (Boolean algebras with operators) of modal logic. Let $\mathcal{P}: \mathbf{Sets} \to \mathbf{Sets}$ be the (covariant) powerset functor. A coalgebra $c: X \to \mathcal{P}(X)$ corresponds to a binary relation $\to$ on X, via $x \to x' \Leftrightarrow x' \in c(x)$. Such a coalgebra is often called a *Kripke structure* or a *frame*. A coalgebra morphism $\big(X \xrightarrow{c} \mathcal{P}(X)\big) \xrightarrow{f} \big(Y \xrightarrow{d} \mathcal{P}(Y)\big)$ corresponds to what is called a "bounded morphism". It satisfies:

(i) $x \to x' \Rightarrow f(x) \to f(x')$
(ii) $f(x) \to y \Rightarrow \exists x' \in X.\, x \to x'$ and $f(x') = y$.

Thus the category of Kripke structures may be identified with the category of coalgebras $\mathbf{CoAlg}(\mathcal{P})$. A transition relation $c\colon X \to \mathcal{P}(X)$ induces a modal operator $\Box_c\colon \mathcal{P}(X) \to \mathcal{P}(X)$ on the Boolean algebra of predices on the state space X. It is given by

$$\Box_c(\alpha) \quad = \quad \{x \in X \mid \forall x'.\, x \to x' \Rightarrow x' \in \alpha\}$$

It is not hard to see that $\Box_c$ preserves (arbitrary) meets.

Let us say that a *Boolean algebra with operators of type* $\mathcal{P}$ consists of a Boolean algebra B with a unary modal operator $\Box\colon B \to B$ preserving finite meets $\top, \wedge$. A homomorphism of such $\mathcal{P}$-BAOs is a homomorphism of Boolean algebras commuting with the $\Box$'s. This yields a category $\mathbf{BAO}(\mathcal{P})$.

The above definition of $\Box_c$ yields a functor $\mathcal{P}\colon \mathbf{CoAlg}(\mathcal{P})^{\mathrm{op}} \to \mathbf{BAO}(\mathcal{P})$. Taking the spectrum (set of ultrafilters) yields a functor $\mathbf{BAO}(\mathcal{P}) \to \mathbf{CoAlg}(\mathcal{P})^{\mathrm{op}}$ in the opposite direction, where the coalgebra structure $c(B)\colon \mathsf{spec}\, B \to \mathcal{P}(\mathsf{spec}\, B)$ induced by a $\mathcal{P}$-BAO $(B, \Box)$ is defined as

$$c(B)(U) \quad = \quad \{V \in \mathsf{spec}\, B \mid \Box^{-1}(U) \subseteq V\}.$$

This yields the following "almost adjoint" situation:

$$\begin{tikzcd} \mathbf{BAO}(\mathcal{P}) \arrow[d, bend right, "\mathsf{spec}"'] \\ \mathbf{CoAlg}(\mathcal{P})^{\mathrm{op}} \arrow[u, bend right, "\mathcal{P}"'] \end{tikzcd} \qquad \text{with} \qquad \mathrm{Id} \overset{\eta}{\Longrightarrow} \mathcal{P}\mathsf{spec} \tag{3}$$

It is not completely trivial that spec is a functor. For a morphism of $\mathcal{P}$-BAOs $h\colon (B_1, \Box_1) \to (B_2, \Box_2)$ we have to prove that $h^{-1}\colon \mathsf{spec}\, B_2 \to \mathsf{spec}\, B_1$ satisfies the above conditions (i) and (ii). The first point is easy, and so we concentrate on the second. Assume $h^{-1}(U) \to_1 W$ and consider the filters $\Box_2^{-1}(U)$ and $h(W)$ of B_2. We claim that the generated filter $\uparrow\!(\Box_2^{-1}(U) \cup h(W))$ does not contain $\perp$. Because, otherwise, there are $b \in \Box_2^{-1}(U)$ and $c \in W$ with $\perp = b \wedge h(c)$; then $b \leq h(\neg c)$ so that $\Box_2(b) \leq h(\Box_1(\neg c))$ and thus $\Box_1(\neg c) \in h^{-1}(U)$; the assumption $h^{-1}(U) \to_1 W$ now yields $\neg c \in W$, contradicting $c \in W$. The claim allows us to use Lemma 2.1, which yields an ultrafilter V of B_2 with $\Box_2^{-1}(U), h(W) \subseteq V$. The first inclusion yields $U \to_2 V$, and the second one yields $W \subseteq h^{-1}(V)$. But then also $h^{-1}(V) \subseteq W$, since if $b \notin W$, then $\neg b \in W$ so that $\neg h(b) = h(\neg b) \in V$, and thus $b \notin h^{-1}(V)$.

The "unit" maps $\eta_B\colon B \to \mathcal{P}(\mathsf{spec}\, B)$ are given by $b \mapsto \{U \mid b \in U\}$. They are homomorphisms of BAOs since

$$\begin{array}{rcl} \big(\Box_{c(B)} \circ \eta_B\big)(b) & = & \{U \mid \forall V \in \mathsf{spec}\, B.\, U \to V \Rightarrow V \in \eta(b)\} \\ & = & \{U \mid \forall V \in \mathsf{spec}\, B.\, \Box^{-1}(U) \subseteq V \Rightarrow b \in V\} \\ & = & \{U \mid b \in \bigcap\{V \in \mathsf{spec}\, B \mid \Box^{-1}(U) \subseteq V\}\} \\ & = & \{U \mid b \in \Box^{-1}(U)\}, \quad \text{by Lemma 2.2} \\ & = & \{U \mid \Box(b) \in U\} \\ & = & (\eta_B \circ \Box)(b). \end{array}$$

For a coalgebra $c: X \to \mathcal{P}(X)$, there is the canonical embedding $\varepsilon: X \to \mathsf{spec}\,\mathcal{P}(X)$ into the "ultrafilter extension". It does not lead to an adjunction in (3): it is folklore knowledge that ε is in general not a morphism of coalgebras (or Kripke structures): the above condition (i) holds, but (ii) is problematic. However, in the special case where each set $c(x)$ is *finite* (so that the level of non-determinism is limited), then the second condition does hold: if $\varepsilon(x) \to V$, then $\Box_c^{-1}(\varepsilon(x)) \subseteq V$. In particular, $c(x) \in V$, and since we can write $c(x)$ as *finite* union of singletons $\{y\}$ for $y \in c(x)$, we get $\{y\} \in V$ for some $y \in c(x)$. It is not hard to show that V is then equal to $\varepsilon(y)$.

Usually in modal logic, see [Lem66, Gol93], the above situation (3) is skipped, and one directly moves to an equivalence, using so-called descriptive Kripke structures, for which the canonical embedding ε is forced to be a homomorphism and a bijection.

In the sequel we limit ourselves to polynomial functors which do not contain the powerset functor $\mathcal{P}$. Then we can obtain a proper adjunction between coalgebras and Boolean algebras with operators, see Theorem 8.4.

5 An elaborate example

This section will concentrate on one example. It involves a bounded stack, specified with coalgebraic operations and associated modal assertions. We will show how stack coalgebras (interpreting the operations) give rise to certain Boolean algebras with operations (BAOs), consisting of the predicates on their state spaces with appropriate nexttime operators (like in [Jac99, Röß00, Röß00, Kur00]). These BAOs in turn can be transformed into stack-coalgebras, in which the original coalgebras can be embedded. This illustrates an essential part of the adjunction between BAOs and coalgebras, which will be described more generally in the next few sections.

The coalgebraic specification of the bounded stacks that we concentrate on will be given in two parts: one for the operations (also called methods), and one for the assertions. The methods are described in Figure 1. The style of specification is as in the experimental language CCSL[2] (for Coalgebraic Class Specification Language), see [HHJT98, Jac00].

BEGIN BoundedStack[D : TYPE, N : ℕ]
  METHOD
    $\mathsf{size}: \mathsf{Self} \longrightarrow \{0, 1, \ldots, N\}$
    $\mathsf{push}: \mathsf{Self} \times D \longrightarrow \mathsf{Self} + \mathsf{Self}$
    $\mathsf{pop}: \mathsf{Self} \longrightarrow \mathsf{Self} + (D \times \mathsf{Self})$
  ASSERTION
    [See Figure 2]
END BoundedStack

Figure 1: Methods of the bounded stack specification

[2]Although the specific modal operators that we use here are not generated automatically by the LOOP compiler for CCSL.

We shall explain the types of these methods. First of all, the type Self is used for the state space (set of states) of the system that is being described. It is considered as a black box, about which nothing is known: all the information we can get about states should come via (possibly repeated) method applications. The size method tells about a particular state in Self what the number of elements is of this state (of the stack). This number is restricted to be an element of the set $\{0, 1, \ldots, N\}$, where N is the maximal size, given as parameter in the specification. The other parameter is the type D, describing the type of the elements that can be stored in the stack. The push method takes a state $x \in \mathsf{Self}$ and an element $d \in D$ and produces an element $\mathsf{push}(x, d)$ which is either of the form $\kappa_1(y)$ or of the form $\kappa_2(z)$ with $y, z \in \mathsf{Self}$. The idea—which will be made explicit by the assertions in Figure 2—is that $\mathsf{push}(x, d) = \kappa_1(y)$ when the push operation fails because the stack is full (has size N), and then $y = x$. The other case, when $\mathsf{push}(x, d) = \kappa_2(z)$, happens when the push operation succeeds because the stack is not full, and in that case the state z is x with d as its top element. Similarly, the result $\mathsf{pop}(x)$ may be of the form $\kappa_1(y)$, or of the form $\kappa_2(d, z)$, with $y, z \in \mathsf{Self}$ and $d \in D$. The first case happens when the pop operation fails because the stack is empty (has size 0), and then $y = x$. In the second case pop succeeds because the stack is non-empty, and the result $\kappa_2(d, z)$ consists of the top element d, and the state z that results from removing d from x. Notice how the (coproduct) types are used explicitly to capture various cases. This is typically coalgebraic. Other (algebraic) approaches use for example subsorts for similar distinctions.

Figure 1 describes what is usually called the interface of the specification, consisting of the methods with their types. This interface gives a lot of information. In a coalgebraic analysis, one first extracts the *interface type* (or *functor*) from this specification. In this case it is:

$$T(X) = (\{0, 1, \ldots, N\} \times (X + X)^D) \times (X + (D \times X)). \tag{4}$$

This type is obtained by moving the input type D of the push method to the right-hand-side of the arrow (using Currying), and by taking the Cartesian product of the resulting output types[3]. Notice that X is used as a place holder for the state space.

A *coalgebra* for this interface functor T consists of an actual set X together with a function $c: X \to T(X)$. Such a function conveniently combines interpretations of the three stack methods size, push and pop in a single map. Indeed, given c, we may extract interpretations $[\![-]\!]$ of these methods as:

$$\begin{array}{rcl} [\![\mathsf{size}]\!](x) & = & \pi_1(\pi_1(c(x))) \\ [\![\mathsf{push}]\!](x, d) & = & \pi_2(\pi_1(c(x)))(d) \\ [\![\mathsf{pop}]\!](x) & = & \pi_2(c(x)) \end{array} \tag{5}$$

Such a coalgebra may thus be seen as a model of the interface. A model of the whole specification (including the assertions) is a model of the interface satisfying the assertions. Often we omit the interpretation brackets $[\![-]\!]$ and identify the coalgebra c with a 3-tuple of functions size, push and pop as above.

[3]Notice that we have grouped this product in a particular way, so that we can restrict ourselves to binary products. But this is an inessential detail.

5.1 Modal operators

The interpretations of assertions are predicates on the state space X of a coalgebra $c\colon X \to T(X)$. Therefore, the powerset $\mathcal{P}(X)$ of subsets of (or predicates on) X plays a distinguished rôle. Associated with a coalgebra c of the stack interface type (4) there are four kinds of (unary) "nexttime" operators on $\mathcal{P}(X)$, corresponding to the four occurrences of X in $T(X)$. These nexttime operators, which we call $\mathsf{PushF}(d)$, $\mathsf{PushS}(d)$, PopF, PopS—with 'F for fail and 'S' for success—are defined for $d\colon D$ and $\alpha \subseteq X$ as:

$$\begin{array}{rcl}
\mathsf{PushF}(d)(\alpha) & = & \{x \in X \mid \forall y \in X.\, \mathsf{push}(x,d) = \kappa_1(y) \Rightarrow \alpha(y)\} \\
\mathsf{PushS}(d)(\alpha) & = & \{x \in X \mid \forall y \in X.\, \mathsf{push}(x,d) = \kappa_2(y) \Rightarrow \alpha(y)\} \\
\mathsf{PopF}(\alpha) & = & \{x \in X \mid \forall y \in X.\, \mathsf{pop}(x) = \kappa_1(y) \Rightarrow \alpha(y)\} \\
\mathsf{PopS}(\alpha) & = & \{x \in X \mid \forall y \in X.\, \forall d \in D.\, \mathsf{pop}(x) = \kappa_2(d,y) \Rightarrow \alpha(y)\}
\end{array}$$

Thus, the predicate $\mathsf{PushF}(d)(\alpha)$ holds for those states x for which it is the case that if pushing d fails (because the stack is full), then α holds for the successor states. Similarly for the other operators. In this way we are turning the set of predicates $\mathcal{P}(X)$ on the state space of the coalgebra c into a Boolean algebra with modal operators. It is not hard to see that the operators $\mathsf{PushF}(d)$, $\mathsf{PushS}(d)$, PopF and PopS preserve meets and non-empty joins of the poset $\mathcal{P}(X)$.

Apart from modal operators there are also observer operations associated with a stack coalgebra. These are, for $n\colon \mathbb{N}$ and $d\colon D$,

$$\begin{array}{rcl}
\mathsf{Size}(n) & = & \{x \in X \mid \mathsf{size}(x) = n\} \\
\mathsf{Top}(d) & = & \{x \in X \mid \forall y \in X.\, \forall e \in D.\, \mathsf{pop}(x) = \kappa_2(e,y) \Rightarrow e = d\}.
\end{array}$$

There are certain "axioms" associated with these operators and observers. They are expressed using exclusive disjunction $\dot{\vee}$ in the Boolean algebra $\mathcal{P}(X)$ of predicates on the state space:

$$\begin{array}{c}
\mathsf{PushF}(d)(\bot) \mathbin{\dot{\vee}} \mathsf{PushS}(d)(\bot) \\
\mathsf{PopF}(\bot) \mathbin{\dot{\vee}} \mathsf{PopS}(\bot) \\
\dot{\bigvee}_{n \in \{0,1,\ldots,N\}} \mathsf{Size}(n) \\
\neg\mathsf{PopS}(\bot) \supset \dot{\bigvee}_{d \in D} \mathsf{Top}(d)
\end{array} \tag{6}$$

The first of these says that either the fail-push or the success-push fails, but not both. And the third one expresses that exactly one natural number can be observed as size.

At this stage one can form a modal logic with elements $\mathsf{Size}(n)$ and $\mathsf{Top}(d)$ as atomic propositions, and with $\neg$, $\wedge$ and $\mathsf{PushF}(d)$, $\mathsf{PushS}(d)$, PopF, PopS, as proposition formers. Actually, for the specification of our bounded stacks we shall also use universal quantification $\forall$, in the form of quantification over the parameter type D, over (a subset of) $\{0, 1, \ldots N\}$ and over the type Prop of propositions. Below we shall often require that types like D are finite, so that quantification over D and over $\{0, 1, \ldots, N\}$ can be replaced by a finite conjunction. But the quantification over Prop means that we are working in *higher order propositional modal logic*. This is needed because bisimilarity of two states can be expressed by saying that they satisfy the same assertions, see [Röß00, Röß99].

Here we do not wish to make this logic that we use a topic of research in itself: we merely want to illustrate the possible use of the modal operators and observers in coalgebraic specification of classes, see Figure 2.

BEGIN BoundedStack[D : TYPE, N : $\mathbb{N}$]
  METHOD
    [See Figure 1]
  ASSERTION
    PushS_non_occurrence : $[\,\forall d \in D.\, \mathsf{Size}(N) \supset \mathsf{PushS}(d)(\bot)\,]$
    PushF_non_occurrence : $[\,\forall d \in D.\, \neg\mathsf{Size}(N) \supset \mathsf{PushF}(d)(\bot)\,]$
    PopS_non_occurrence : $[\,\mathsf{Size}(0) \supset \mathsf{PopS}(\bot)\,]$
    PopF_non_occurrence : $[\,\neg\mathsf{Size}(0) \supset \mathsf{PopF}(\bot)\,]$
    no_change_PushF : $[\,\forall d \in D.\, \forall \alpha \in \mathsf{Prop}.\, \mathsf{Size}(N) \supset (\mathsf{PushF}(d)(\alpha) \supset\subset \alpha)\,]$
    Size_PushS : $[\ \forall d \in D.\, \forall n \in \{0, 1, \ldots, N-1\}.$
        $\mathsf{Size}(n) \supset \mathsf{PushS}(d)(\mathsf{Size}(n+1))\,]$
    Top_PushS : $[\,\forall d \in D.\, \mathsf{PushS}(d)(\mathsf{Top}(d))\,]$
    Pop_PushS : $[\,\forall d \in D.\, \forall \alpha \in \mathsf{Prop}.\, \neg\mathsf{Size}(N) \supset (\mathsf{PushS}(d)(\mathsf{PopS}(\alpha)) \supset\subset \alpha)\,]$
    no_change_PopF : $[\,\forall \alpha \in \mathsf{Prop}.\, \mathsf{Size}(0) \supset (\mathsf{PopF}(\alpha) \supset\subset \alpha)\,]$
    Size_PopS : $[\,\forall n \in \{0, 1, \ldots, N-1\}.\, \mathsf{Size}(n+1) \supset \mathsf{PopS}(\mathsf{Size}(n))\,]$
END BoundedStack

Figure 2: Assertions of the bounded stack specification

As usual in formal specification, the assertions have names, which are written as "assertion_name: [predicate on Self]". The first assertion 'PushS_non_occurrence' expresses that if the stack is full, a succesfull push will not happen, *i.e.* the second coproduct option in the output type $\mathsf{Self} + \mathsf{Self}$ of the push method does not occur. Thus, when the stack is full, the fail-push operator PushF applies. The assertion 'no_change_PushF' then says that the resulting state (in the first coproduct option) satisfies the same predicates as the original state.

A *model* of the bounded stack specification in Figure 2 is a coalgebra $c\colon X \to T(X)$ of the interface type (4) of the associated signature, such that all the assertions, when interpreted as predicates on the state space X, are true for all $x \in X$. An obvious model can be constructed with as state space the set $\mathsf{List}(D, N)$ of lists of elements of length at most N. Explicitly,

$$\mathsf{List}(D, N) = \coprod_{0 \le n \le N} D^n$$

with T-coalgebra $c\colon \mathsf{List}(D, N) \to (\{0, 1, \ldots, N\} \times (\mathsf{List}(D, N) + \mathsf{List}(D, N))^D) \times (\mathsf{List}(D, N) + (D \times \mathsf{List}(D, N)))$ defined by forming a nested tuple of the size, push and pop operations on this state space:

$$\begin{aligned} c(\langle d_1, \ldots d_n\rangle) = \ \ & ((n, \\ & \lambda d \in D.\, \text{if } n = N \text{ then } \kappa_1(\langle d_1, \ldots d_n\rangle) \text{ else } \kappa_2(\langle d, d_1, \ldots d_n\rangle)), \\ & \text{if } n = 0 \text{ then } \kappa_1(\langle d_1, \ldots d_n\rangle) \text{ else } \kappa_2(d_1, \langle d_2, \ldots d_n\rangle)) \end{aligned}$$

It is easy to see that the assertions from Figure 2 hold for this coalgebra.

5.1. Remark. The above modal operators $\mathsf{PushF}(d)$, $\mathsf{PushS}(d)$, PopF and PopS are the so-called *weak* nexttime operators because, they are of the form "for all such-and-such successor states ...", see [Jac99]. The associated *strong* nexttime operators are obtained by negating twice, as in $\neg\mathsf{PopF}(\neg\alpha) = \{x \mid \exists y.\, \mathsf{pop}(x) = \kappa_1(y) \wedge y \in \alpha\}$. They express that there actually is a successor state with a certain property. The latter are used in [Röß99].

Regarding the observers, we have used several functions, namely Size and Top. More succintly, one can use elements in $T(1)$ as observations for the functor T, with a single "valuation function" $T(1) \to \mathcal{P}(X)$ given by $a \mapsto \{x \mid T(!)(c(x)) = a\}$, see [Jac99]–where ! is the unique function $X \to 1$. But the approach with Size and Top is more convenient in concrete specifications.

5.2 A state space of ultrafilters

At this stage we assume we have an arbitrary coalgebra $c\colon X \to T(X)$ of the stack interface type (4) interpreting the stack methods, without assuming that it forms a model of the stack assertions. As we have seen in the previous subsection, such a coalgebra gives rise to a Boolean algebra $\mathcal{P}(X)$ with modal operators $\mathsf{PushF}(d)$, $\mathsf{PushS}(d)$, PopF, PopS on $\mathcal{P}(X)$ and observers $\mathsf{Size}\colon \{0, 1, \ldots, N\} \to \mathcal{P}(X)$, $\mathsf{Top}\colon D \to \mathcal{P}(X)$. Below we shall describe a construction which turns the set $\mathsf{spec}\,\mathcal{P}(X)$ of ultrafilters of $\mathcal{P}(X)$ into a state space for a new T-coalgebra, in which the original state space X can be embedded via the function $\varepsilon_X\colon X \hookrightarrow \mathsf{spec}\,\mathcal{P}(X)$ given by $\varepsilon_X(x) = \{\alpha \in \mathcal{P}(X) \mid x \in \alpha\}$. This "ultrafilter extension" will be described in more general terms in later sections.

Now we assume that the parameter D for the stack data in Figures 1 and 2 is finite. The stack axioms (6) will play an important rôle. For example, for an arbitrary ultrafilter $U \in \mathsf{spec}\,\mathcal{P}(X)$ we have $\mathsf{PushF}(d)(\bot) \mathbin{\dot\vee} \mathsf{PushS}(d)(\bot) = \top \in U$. Therefore either $\mathsf{PushF}(\bot) \in U$ or $\mathsf{PushS}(d)(\bot) \in U$, but not both. Similarly, if $\mathsf{Push}(d)(\bot) \notin U$, then $\mathsf{Top}(d) \in U$, for exactly one element $d \in D$. Here we use that D is finite, so that we have a finite (exclusive) disjunction over D in (6).

We shall now put a coalgebra structure $\mathsf{spec}\,\mathcal{P}(X) \to T(\mathsf{spec}\,\mathcal{P}(X))$ by defining individual functions $\mathsf{size}\colon \mathsf{spec}\,\mathcal{P}(X) \to \mathbb{N}$, $\mathsf{push}\colon \mathsf{spec}\,\mathcal{P}(X) \times D \to (\mathsf{spec}\,\mathcal{P}(X)) + (\mathsf{spec}\,\mathcal{P}(X))$ and $\mathsf{pop}\colon \mathsf{spec}\,\mathcal{P}(X) \to (\mathsf{spec}\,\mathcal{P}(X)) + (D \times \mathsf{spec}\,\mathcal{P}(X))$ as follows.

$$
\begin{array}{rcl}
\mathsf{size}(U) & = & \text{the unique } m \text{ with } \mathsf{Size}(m) \in U \\
 & & (\text{using } \dot{\bigvee}_{n \in \{0,1,\ldots,N\}} \mathsf{Size}(n) = \top \in U) \\
\mathsf{push}(U, d) & = & \begin{cases} \kappa_1(\mathsf{PushF}(d)^{-1}(U)) & \text{if } \neg\mathsf{PushF}(d)(\bot) \in U \\ \kappa_2(\mathsf{PushS}(d)^{-1}(U)) & \text{if } \neg\mathsf{PushS}(d)(\bot) \in U \end{cases} \\
\mathsf{pop}(U) & = & \begin{cases} \kappa_1(\mathsf{PopF}^{-1}(U)) & \text{if } \neg\mathsf{PopF}(\bot) \in U \\ \kappa_2(d, \mathsf{PopS}^{-1}(U)) & \text{if } \neg\mathsf{PopS}(\bot) \in U \text{ and } \mathsf{Top}(d) \in U. \end{cases}
\end{array}
$$

Notice that if $\neg\mathsf{PushF}(d)(\bot) \in U$, then the set $\mathsf{PushF}(d)^{-1}(U)$ is an ultrafilter by Lemma 2.3. Hence these functions are well-defined.

The next result summarises some points about the ultrafilter extension for stack coalgebras. They will be generalised in the rest of this paper.

5.2. Proposition. (i) *The embedding $\varepsilon_X\colon X \to \mathsf{spec}\,\mathcal{P}(X)$ is a homomorphism of coalgebras.*

(ii) *The above construction (coalgebra on X) $\mapsto$ (coalgebra on $\mathsf{spec}\,\mathcal{P}(X)$) extends to a functor $\mathsf{spec}\ \mathcal{P}$: $CoAlg(T) \to CoAlg(T)$, and ε is a natural transformation from the identity functor $CoAlg(T) \to CoAlg(T)$ to $\mathsf{spec}\ \mathcal{P}$.*

(iii) *The original coalgebra $c\colon X \to T(X)$ satisfies the stack assertions from Figure 2, if and only if the induced coalgebra on $\mathsf{spec}\,\mathcal{P}(X)$ satisfies them.*

Proof. (i) The first thing to note is that $\mathsf{Size}(n) \in \varepsilon(x) \Leftrightarrow x \in \mathsf{Size}(n) \Leftrightarrow \mathsf{size}(x) = n$. Therefore $\mathsf{size}(\varepsilon_X(x)) = \mathsf{size}(x)$, where $\mathsf{size}(x)$ is defined as in the equations (5).

Similarly, $\neg\mathsf{PushF}(d)(\bot) \in \varepsilon(x) \Leftrightarrow x \notin \mathsf{PushF}(d)(\bot) \Leftrightarrow \mathsf{push}(x, d) = \kappa_1(x')$, for some (unique) $x' \in X$. In that case ε commutes with push:

$$\begin{array}{rcl}
\mathsf{push}(\varepsilon_X(x), d) & = & \kappa_1(\{\alpha \mid \mathsf{PushF}(d)(\alpha) \in \varepsilon_X(x)\}) \\
& = & \kappa_1(\{\alpha \mid x \in \mathsf{PushF}(d)(\alpha)\}) \\
& = & \kappa_1(\{\alpha \mid \forall y.\, \mathsf{size}(x, d) = \kappa_1(y) \Rightarrow y \in \alpha\}) \\
& = & \kappa_1(\varepsilon_X(x')).
\end{array}$$

The other cases are left to the interested reader.

(ii) Assume we have a homomorphism of coalgebras $f\colon X \to Y$ from $(c\colon X \to T(X))$ to $(d\colon Y \to T(Y))$. It is not hard to show that the induced function $\mathsf{spec}\,\mathcal{P}(X) \to \mathsf{spec}\ \mathcal{P}(Y)$ given by $U \mapsto \{\beta \in \mathcal{P}(Y) \mid f^{-1}(\beta) \in U\}$ is again a homomorphism. This is because f^{-1} commutes with the modal operators and observers. For instance, $f^{-1}(\mathsf{Size}(n)) = \mathsf{Size}(n)$ and $f^{-1}(\mathsf{PushF}(d)(\beta)) = \mathsf{PushF}(d)(f^{-1}(\beta))$. Also, the diagram

$$\begin{tikzcd}
X \arrow[r] \arrow[d, "\varepsilon_X"'] & Y \arrow[d, "\varepsilon_Y"] \\
\mathsf{spec}\,\mathcal{P}(X) \arrow[r] & \mathsf{spec}\ \mathcal{P}(Y)
\end{tikzcd}$$

(standardly) commutes, making ε a natural transformation.

(iii) The point is to show for each assertion φ and ultrafilter $U \subseteq \mathcal{P}(X)$ that

$$U \in [\![\, \varphi \,]\!]^{\mathsf{spec}\,\mathcal{P}(X)} \iff [\![\, \varphi \,]\!]^X \in U$$

As illustration, we prove this equivalence for the assertions 'PushS_non_occurrence' and 'no_change_PushF'. For convenience, we omit the interpretation brackets $[\![-]\!]$.

$$\begin{array}{l}
U \in \big(\mathsf{Size}(N) \supset \mathsf{PushS}(d)(\bot)\big) \\
\quad\Leftrightarrow\quad U \in \mathsf{Size}(N) \Rightarrow U \in \mathsf{PushS}(d)(\bot) \\
\quad\Leftrightarrow\quad \mathsf{size}(U) = N \Rightarrow \forall V.\, \mathsf{push}(U, d) = \kappa_2(V) \Rightarrow V \in \bot \\
\quad\Leftrightarrow\quad \mathsf{Size}(N) \in U \Rightarrow \mathsf{PushS}(d)(\bot) \in U \\
\quad\Leftrightarrow\quad \mathsf{Size}(N) \supset \mathsf{PushS}(d)(\bot) \in U
\end{array}$$

And also:

$$
\begin{array}{ll}
\multicolumn{2}{l}{U \in \big(\mathsf{Size}(N) \supset (\mathsf{PushF}(d)(A) \supset\!\subset A)\big) \quad \text{for all } A \subseteq \mathsf{spec}\,\mathcal{P}(X)} \\
\Leftrightarrow & U \in \mathsf{Size}(N) \Rightarrow (U \in \mathsf{PushF}(d)(A) \Leftrightarrow U \in A) \quad \text{for all } A \\
\Leftrightarrow & \mathsf{size}(U) = N \Rightarrow U \in \mathsf{PushF}(d)(\{U\}) \\
\Leftrightarrow & \mathsf{Size}(N) \in U \Rightarrow (\neg\mathsf{PushF}(d)(\bot) \in U \Rightarrow \mathsf{PushF}(d)^{-1}(U) = U) \\
\Leftrightarrow & \mathsf{Size}(N) \in U \Rightarrow (\mathsf{PushF}(d)(\alpha) \in U \Leftrightarrow \alpha \in U) \quad \text{for all } \alpha \subseteq X \\
 & \quad \text{Since we proved above: } \mathsf{Size}(N) \in U \text{ implies } \mathsf{PushS}(d)(\bot) \in U, \\
 & \quad \text{and so } \neg\mathsf{PushF}(d)(\bot) \in U \text{ since } \mathsf{PushF}(d)(\bot) \mathbin{\dot{\vee}} \mathsf{PushS}(d)(\bot) = \top \\
\Leftrightarrow & \mathsf{Size}(N) \in U \Rightarrow \mathsf{PushF}(d)(\alpha) \supset\!\subset \alpha \in U \quad \text{for all } \alpha \\
\Leftrightarrow & \mathsf{Size}(N) \supset (\mathsf{PushF}(d)(\alpha) \supset\!\subset \alpha) \in U \quad \text{for all } \alpha. \qquad \square
\end{array}
$$

Notice that the embedding ε into the ultrafilter extension is a homomorphism of coalgebras—unlike for Kripke structures, see the end of Section 4. The third point of this result involving formulas will not be further elaborated in (the current version of) this paper.

6 Coalgebras, operators and observers

This section will explain the basic techniques for obtaining for an arbitrary polynomial functor modal operators and observers as used for stack coalgebras in the previous section. These techniques involve paths and predicate lifting, much like in [Jac99].

Let T be a polynomial functor which contains another polynomial functor S as constituent, like in:

$$T = \boxed{\;\cdots\; S \;\cdots\;}$$

We shall make such occurrences explicit by defining how such an S can be reached via a "path" p inside T. In that case we write $T \overset{p}{\rightsquigarrow} S$. The path p is a finite list of symbols π_1, π_2, κ_1, κ_2, $\mathrm{ev}(d)$, for elements $d \in D$ of sets D occurring as exponent in T. Such a path tells us how to find S in T.

6.1. Definition. (i) Let T and S be polynomial functors. The relation $T \overset{p}{\rightsquigarrow} S$ is the least relation generated by the following clauses.

- $T \overset{\langle\rangle}{\rightsquigarrow} T$, where $\langle\rangle$ is the empty list.
- $T_1 \times T_2 \overset{\pi_1 \cdot p}{\rightsquigarrow} S$ for $T_1 \overset{p}{\rightsquigarrow} S$, and $T_1 \times T_2 \overset{\pi_2 \cdot p}{\rightsquigarrow} S$ for $T_2 \overset{p}{\rightsquigarrow} S$.
- $T_1 + T_2 \overset{\kappa_1 \cdot p}{\rightsquigarrow} S$ for $T_1 \overset{p}{\rightsquigarrow} S$, and $T_1 + T_2 \overset{\kappa_2 \cdot p}{\rightsquigarrow} S$ for $T_2 \overset{p}{\rightsquigarrow} S$.
- $T_1^D \overset{\mathrm{ev}(d) \cdot p}{\rightsquigarrow} S$ for all $d \in D$ and $T_1 \overset{p}{\rightsquigarrow} S$.

(ii) For a polynomial functor T we define the sets of modal operators $Op(T)$ and observers $Ob(T)$ as:

$$Op(T) = \{p \mid T \overset{p}{\rightsquigarrow} \mathrm{Id}\} \qquad \text{and} \qquad Ob(T) = \{p \mid T \overset{p}{\rightsquigarrow} D\}.$$

Sometimes we combine these operators and observers in a set

$$OpOb(T) = Op(T) \cup Ob(T).$$

Basically we are only interested in the operator and observer paths, having identity and constant functors as targets. But it is convenient to define paths with more general target functors, for instance to do (loaded) induction proofs.

It is not hard to see that these paths can be composed (via concatenation of lists): if $T_1 \overset{p}{\rightsquigarrow} T_2$ and $T_2 \overset{q}{\rightsquigarrow} T_3$, then $T_1 \overset{p\cdot q}{\rightsquigarrow} T_3$.

6.2. Definition. (i) For a path $T \overset{p}{\rightsquigarrow} S$ and an arbitrary set X there is a "predicate lifting" (see [HJ98, Jac99]) function

$$\mathcal{P}(S(X)) \xrightarrow{(-)^p} \mathcal{P}(T(X))$$

defined on $\alpha \subseteq S(X)$ by induction on p:

- $\alpha^{\langle\rangle} \quad = \quad \alpha$
- $$\begin{aligned} \alpha^{\pi_1 \cdot p} &= \{z \mid \pi_1(z) \in \alpha^p\} \\ \alpha^{\pi_2 \cdot p} &= \{z \mid \pi_2(z) \in \alpha^p\} \end{aligned}$$
- $$\begin{aligned} \alpha^{\kappa_1 \cdot p} &= \{z \mid \forall y.\, z = \kappa_1(y) \Rightarrow y \in \alpha^p\} \\ \alpha^{\kappa_2 \cdot p} &= \{z \mid \forall y.\, z = \kappa_2(y) \Rightarrow y \in \alpha^p\} \end{aligned}$$
- $\alpha^{\mathrm{ev}(d)\cdot p} \quad = \quad \{f \mid f(d) \in \alpha^p\}.$

(ii) For a coalgebra $c: X \to T(X)$ we define for each modal operator $(T \overset{p}{\rightsquigarrow} \mathrm{Id}) \in Op(T)$ an interpretation

$$\mathcal{P}(X) \xrightarrow{[\![p]\!]} \mathcal{P}(X) \quad \text{by} \quad [\![p]\!](\alpha) = c^{-1}(\alpha^p) = \{x \in X \mid c(x) \in \alpha^p\}$$

and for each observer $(T \overset{p}{\rightsquigarrow} D) \in Ob(T)$ an interpretation

$$D \cup \{\bot\} \xrightarrow{[\![p]\!]} \mathcal{P}(X) \quad \text{by} \quad \left\{ \begin{array}{l} [\![p]\!](d) = c^{-1}(\{d\}^p) = \{x \in X \mid c(x) \in \{d\}^p\} \\ [\![p]\!](\bot) = c^{-1}(\bot^p) = \{x \in X \mid c(x) \in \bot^p\} \end{array} \right.$$

It is assumed here that $\bot$ is not an element of D.

It is not hard to check that the stack operators and observers from the previous section arise by following the above definition for the stack interface functor (4). Explicitly,

$$\begin{aligned} \mathsf{PushF}(d) &= [\![\pi_1 \cdot \pi_2 \cdot \mathrm{ev}(d) \cdot \kappa_1]\!] \\ \mathsf{PushS}(d) &= [\![\pi_1 \cdot \pi_2 \cdot \mathrm{ev}(d) \cdot \kappa_2]\!] \\ \mathsf{PopF} &= [\![\pi_2 \cdot \kappa_1]\!] \\ \mathsf{PopS} &= [\![\pi_2 \cdot \kappa_2 \cdot \pi_2]\!] \\ \mathsf{Size} &= [\![\pi_1 \cdot \pi_1]\!] \\ \mathsf{Top} &= [\![\pi_2 \cdot \kappa_2 \cdot \pi_1]\!] \end{aligned}$$

In the stack example we did not use the obervers Size and Top on $\bot$.

The following easy results are worth making explicit.

6.3. Lemma. (i) *The predicate lifting function $(-)^p: \mathcal{P}(S(X)) \to \mathcal{P}(T(X))$ associated with a path $T \overset{p}{\rightsquigarrow} S$ preserves arbitrary meets $\bigwedge$ and non-empty joins $\bigvee$.*

(ii) *The interpretations* $[\![p]\!]\colon \mathcal{P}(X) \to \mathcal{P}(X)$ *of modal operators* $p \in Op(T)$ *preserve arbitrary meets and non-empty joins.*

(iii) *Predicate lifting preserves composition, in the sense that if* $T_1 \overset{p}{\rightsquigarrow} T_2$ *and* $T_2 \overset{q}{\rightsquigarrow} T_3$*, then the following diagram commutes.*

$$\begin{tikzcd}
\mathcal{P}(T_3(X)) \arrow[r, "(-)^q"] \arrow[dr, "(-)^{p\cdot q}"'] & \mathcal{P}(T_2(X)) \arrow[d, "(-)^p"] \\
& \mathcal{P}(T_1(X))
\end{tikzcd}$$

(iv) *Predicate lifting is natural: for an arbitrary function* $f\colon X \to Y$ *and path* $T \overset{p}{\rightsquigarrow} S$ *one has*

$$\begin{tikzcd}
\mathcal{P}(S(X)) \arrow[r, "(-)^p"] & \mathcal{P}(T(X)) \\
\mathcal{P}(S(Y)) \arrow[r, "(-)^p"'] \arrow[u, "S(f)^{-1}"] & \mathcal{P}(T(Y)) \arrow[u, "T(f)^{-1}"']
\end{tikzcd}$$

The preservation of the (all) meets by $(-)^p$ and $[\![p]\!]$ is standard. As a result, these operators have left adjoints. The adjoints are described explicitly in [Jac99]. The preservation of non-empty joins however is special, and is due to the deterministic nature of the coalgebras that we are considering. It no longer holds if powersets are included in the functors T that we consider. Therefore we shall be careful in using this preservation property, and shall always make its use explicit.

As an aside we mention that the properties in the last two points above give rise to a certain indexed categorical structure. Let us write **PFP** for the category of polynomial functors and paths between them. Each set X yields an "indexed category" $\overline{X}\colon \mathbf{PFP}^{\mathrm{op}} \to \mathbf{PoSets}$ by $T \mapsto \mathcal{P}(T(X))$ and $(T \overset{p}{\rightsquigarrow} S) \mapsto (-)^p$. The assignment $X \mapsto \overline{X}$ also yields a functor $\mathbf{Sets}^{\mathrm{op}} \to [\mathbf{PFP}^{\mathrm{op}}, \mathbf{PoSets}]$ to a functor category: a function $f\colon X \to Y$ yields a natural transformation $\overline{Y} \Rightarrow \overline{X}$ with component at T given by $T(f)^{-1}$.

Proof. (i) By induction on paths. The most interesting case is preservation of non-empty joins for a coproduct functor:

$$\begin{array}{rcl}
\left(\bigvee_{i\in I} \alpha_i\right)^{\kappa_1\cdot p} & = & \{z \mid \forall y.\, z = \kappa_1(y) \Rightarrow \exists i \in I.\, y \in (\alpha_i)^p\} \\
& \overset{*}{=} & \{z \mid \exists i \in I.\, \forall y.\, z = \kappa_1(y) \Rightarrow y \in (\alpha_i)^p\} \\
& = & \bigvee_{i\in I} (\alpha_i)^p,
\end{array}$$

where the inclusion $(\subseteq)$ of the marked equation $\overset{*}{=}$ is obtained as follows. Given z, with $\forall y.\, z = \kappa_1(y) \Rightarrow \exists i \in I.\, y \in (\alpha_i)^p$, then either $z = \kappa_1(u)$, or $z = \kappa_2(v)$, for certain u, v. In the first case we know that $u \in (\alpha_i)^p$ for some $i \in I$. This i will do, since if $z = \kappa_1(u')$, then $u' = u$ and thus $u' \in (\alpha_i)^p$. In the second case an arbitrary $i \in I$ does the job.

(ii) By (i), using that inverse image functions c^{-1} preserve arbitrary meets and joins.

(iii) + (iv) By induction on p. $\square$

Note that predicate lifting $(-)^p$ does not preserve all Boolean structure, because it does not preserve $\perp$ (in general). Therefore, the predicates $\perp^p$ (or often their negations $\neg(\perp^p)$) play an important rôle. They tell whether or not there is a result along path p.

6.4. Lemma. *Let T be a polynomial functor and X an arbitrary set.*

(i) *For each observer $(T \overset{p}{\rightsquigarrow} D) \in Ob(T)$, the implication*

$$\neg(\perp^p) \supset \dot{\bigvee}_{d \in D} \{d\}^p$$

holds (i.e. is $\top$) in the poset $\mathcal{P}(T(X))$.

(ii) *For each path $T \overset{p}{\rightsquigarrow} S$, there is a canonical function*

$$\neg(\perp^p) \xrightarrow{\;\widehat{p}\;} S(X)$$

satisfying for $z \in \neg(\perp^p) \subseteq T(X)$ and $\alpha \subseteq S(X)$,

$$\widehat{p}(z) \in \alpha \iff z \in \alpha^p.$$

(iii) *For each path $T \overset{p}{\rightsquigarrow} S$, there is an equality of predicates on $T(X)$ of the form:*

$$\neg(\perp^p) = \bigvee_{q \in OpOb(S)} \neg(\perp^{p \cdot q}).$$

(Of course, we can eliminate the two $\neg$'s and replace $\bigvee$ by $\bigwedge$, but the above formulation is the one that is most useful.)

Proof. (i) By induction on the structure of the functor T.

- The case where T is the identity functor cannot happen, because this functor has no observers.
- If T is a constant functor D, then $Op(T) = \emptyset$ and $Ob(T) = \{\langle\rangle\}$. Thus

$$\neg(\perp^p) \supset \dot{\bigvee}_{d \in D} \{d\}^p = \{e \in D \mid e \notin \perp^{\langle\rangle} \Rightarrow \exists! d \in D.\, e \in \{d\}^{\langle\rangle}\} = \top.$$

- If $T = T_1 \times T_2$ and $p = \pi_1 \cdot q$, say, then for $z \in T_1(X) \times T_2(X)$ with $z \notin \perp^p$, *i.e.* with $\pi_1(z) \notin \perp^q$, we have $\pi_1(z) \in \{d\}^q$ for a unique $d \in D$ by induction hypothesis. But this means $z \in \{d\}^p$. Similarly for $p = \pi_2 \cdot q$.
- If $T = T_1 + T_2$ and $p = \kappa_1 \cdot q$, then for $z \in T_1(X) + T_2(X)$ with $z \notin \perp^p$, we have $z = \kappa_1(y)$ for some $y \in T_1(X)$ with $y \notin \perp^q$. Thus we can apply the induction hypothesis to y, which yields $y \in \{d\}^q$ for a unique $d \in D$. But then $z \in \{d\}^p$.
- If $T = T_1^E$ and $p = \mathrm{ev}(e) \cdot q$ for $e \in E$, then for $f \in T_1(X)^E$ with $f \notin \perp^p$ we have $f(e) \notin \perp^q$, so that by induction hypothesis $f(e) \in \{d\}^q$ for a unique $d \in D$. Then $f \in \{d\}^p$.

(ii) By induction on the path p.

- If $p = \langle\rangle$ and thus $S = T$, then $\neg(\perp^p) = \neg\perp = \top = T(X)$ so we take $\widehat{p} = \mathrm{id}\colon T(X) \to S(X)$. It clearly satisfies $\widehat{p}(z) \in \alpha \Leftrightarrow z \in \alpha^p$.

- Assume $p = \pi_1 \cdot q$ where $T = T_1 \times T_2$. The induction hypothesis gives us $\widehat{q}\colon \neg(\perp^q) \to S(X)$. For $z \in \neg(\perp^p)$ we have $\pi_1(z) \notin \perp^q$, so we may take $\widehat{p}(z) = \widehat{q}(\pi_1(z))$. Then $\widehat{p}(z) \in \alpha \Leftrightarrow \pi_1(z) \in \alpha^q \Leftrightarrow z \in \alpha^p$.
- If $p = \kappa_1 \cdot q$ for $T = T_1 + T_2$, then we have $\widehat{q}\colon \neg(\perp^q) \to S(X)$ by induction hypothesis. For $z \in \neg(\perp^p)$ we have $z = \kappa_1(y)$ with $y \notin \perp^q$ for a (unique) y. Hence we put $\widehat{p}(z) = \widehat{q}(y)$. Then $\widehat{p}(z) \in \alpha \Leftrightarrow y \in \alpha^q \Leftrightarrow z \in \alpha^p$.
- If $p = \mathrm{ev}(e) \cdot q$ for $e \in E$ in $T = T_1^E$, then for $z \in \neg(\perp^p)$ we have $z(e) \notin \perp^q$ so we take $\widehat{p}(z) = \widehat{q}(z(e))$. Then: $\widehat{p}(z) \in \alpha \Leftrightarrow z(e) \in \alpha^q \Leftrightarrow z \in \alpha^p$.

(iii) By induction on the structure of S.

- If S is the identity functor or a constant functor the result is obvious.
- If $S = S_1 \times S_2$ then

$$\begin{aligned}
\bigvee_{q \in Op\,Ob(S)} \neg(\perp^{p\cdot q}) &= \Bigg(\bigvee_{q \in Op\,Ob(S_1)} \neg(\perp^{p\cdot\pi_1\cdot q})\Bigg) \vee \Bigg(\bigvee_{q \in Op\,Ob(S_2)} (\perp^{p\cdot\pi_2\cdot q})\Bigg) \\
&\overset{\text{(IH)}}{=} \neg(\perp^{p\cdot\pi_1}) \vee \neg(\perp^{p\cdot\pi_2}) \\
&= \neg\big((\perp^{\pi_1})^p \wedge (\perp^{\pi_2})^p\big) \qquad \text{by Lemma 6.3 (iii)} \\
&= \neg(\perp^{\pi_1} \wedge \perp^{\pi_2})^p \qquad \text{by Lemma 6.3 (i)} \\
&= \neg(\perp^p).
\end{aligned}$$

- If S is a coproduct $S_1 + S_2$, a similar computation can be used because

$$\begin{aligned}
\perp^{\kappa_1} \wedge \perp^{\kappa_2} &= \{z \mid \forall y.\, z = \kappa_1(y) \Rightarrow y \in \perp\} \wedge \{z \mid \forall y.\, z = \kappa_2(y) \Rightarrow y \in \perp\} \\
&= \{z \mid \neg\exists y.\, z = \kappa_1(y)\} \wedge \{z \mid \neg\exists y.\, z = \kappa_2(y)\} \\
&= \perp.
\end{aligned}$$

- If $S = S_1^E$, then

$$\begin{aligned}
\bigvee_{q \in Op\,Ob(S)} \neg(\perp^{p\cdot q}) &= \bigvee_{e \in E} \bigvee_{q \in Op\,Ob(S_1)} \neg(\perp^{p\cdot \mathrm{ev}(e)\cdot q}) \\
&\overset{\text{(IH)}}{=} \bigvee_{e \in E} \neg(\perp^{p\cdot \mathrm{ev}(e)}) \\
&= \neg\Bigg(\bigwedge_{e \in E} (\perp^{\mathrm{ev}(e)})^p\Bigg) \\
&= \neg\Bigg(\bigwedge_{e \in E} \perp^{\mathrm{ev}(e)}\Bigg)^p \\
&= \neg(\perp^p).
\end{aligned}$$

The fact that E is non-empty is used for the last step. □

7 Decomposition and assembly of coalgebras

The previous section concentrated on properties of predicates and paths. These will form in the current section the basis for some elementary results about observers and operators for coalgebras. These results will be axiomatised to get appropriate Boolean algebras with operators in the next section.

7.1. Lemma. *For a coalgebra $c\colon X \to T(X)$ of a polynomial functor T, the subsets $\big(\neg[\![p]\!](\bot) \subseteq X\big)_{p\in OpOb(T)}$ satisfy the following properties.*

(i) *These subsets cover X:*

$$\bigvee_{p\in OpOb(T)} \neg[\![p]\!](\bot) \;=\; \top.$$

(ii) *For each path $T \overset{p}{\rightsquigarrow} T_1 \times T_2$ and $i \in \{1,2\}$,*

$$\bigvee_{q\in OpOb(T_1\times T_2)} \neg[\![p\cdot q]\!](\bot) \;=\; \bigvee_{q\in OpOb(T_i)} \neg[\![p\cdot \pi_i \cdot q]\!](\bot).$$

(iii) *For each path $T \overset{p}{\rightsquigarrow} T_1 + T_2$,*

$$\begin{aligned}&\bigvee_{q\in OpOb(T_1+T_2)} \neg[\![p\cdot q]\!](\bot)\\ &\quad= \left(\bigvee_{q\in OpOb(T_1)} \neg[\![p\cdot \kappa_1 \cdot q]\!](\bot)\right) \dot{\vee} \left(\bigvee_{q\in OpOb(T_2)} \neg[\![p\cdot \kappa_2 \cdot q]\!](\bot)\right).\end{aligned}$$

(iv) *For each path $T \overset{p}{\rightsquigarrow} T_1^E$ and $e\in E$,*

$$\bigvee_{q\in OpOb(T_1^E)} \neg[\![p\cdot q]\!](\bot) \;=\; \bigvee_{q\in OpOb(T_1)} \neg[\![p\cdot ev(e) \cdot q]\!](\bot).$$

(v) *For each observer path $T \overset{p}{\rightsquigarrow} D$,*

$$[\![p]\!](\bot) \supset \bigwedge_{d\in D} [\![p]\!](d) \qquad \textit{and} \qquad \neg[\![p]\!](\bot) \supset \dot{\bigvee}_{d\in D} [\![p]\!](d)$$

Proof. Lemma 6.4 (iii) is crucial in all cases.

(i)
$$\begin{aligned}\bigvee_{p\in OpOb(T)} \neg[\![p]\!](\bot) &= c^{-1}\left(\bigvee_{p\in OpOb(T)} \neg\big(\bot^{p}\big)\right)\\ &= c^{-1}\big(\neg\big(\bot^{\langle\rangle}\big)\big)\\ &= c^{-1}(\top)\\ &= \top.\end{aligned}$$

(ii)
$$\begin{aligned}\bigvee_{q\in OpOb(T_1\times T_2)} \neg[\![p\cdot q]\!](\bot) &= c^{-1}\left(\bigvee_{q\in OpOb(T_1\times T_2)} \neg\big(\bot^{p\cdot q}\big)\right)\\ &= c^{-1}\big(\neg\big(\bot^{p}\big)\big)\\ &= c^{-1}\big(\neg\big(\big(\bot^{\pi_i}\big)^{p}\big)\big)\\ &= c^{-1}\big(\neg\big(\bot^{p\cdot\pi_i}\big)\big)\\ &= c^{-1}\left(\bigvee_{q\in OpOb(T_i)} \neg\big(\bot^{p\cdot\pi_i\cdot q}\big)\right)\\ &= \bigvee_{q\in OpOb(T_i)} \neg[\![p\cdot \pi_i \cdot q]\!](\bot).\end{aligned}$$

(iii) $$\bigvee_{q\in OpOb(T_1+T_2)} \neg[\![p\cdot q]\!](\bot)$$
$$\begin{aligned}
&= c^{-1}\left(\bigvee_{q\in OpOb(T_1+T_2)} \neg\big(\bot^{p\cdot q}\big)\right)\\
&= c^{-1}\big(\neg\big(\bot^{p}\big)\big)\\
&\overset{*}{=} c^{-1}\Big(\neg\big(\bot^{p\cdot\kappa_1}\big)\ \dot\vee\ \neg\big(\bot^{p\cdot\kappa_2}\big)\Big)\\
&= c^{-1}\left(\bigvee_{q\in OpOb(T_1)} \neg\big(\bot^{p\cdot\kappa_1\cdot q}\big)\right)\ \dot\vee\ c^{-1}\left(\bigvee_{q\in OpOb(T_2)} \neg\big(\bot^{p\cdot\kappa_2\cdot q}\big)\right)\\
&= \left(\bigvee_{q\in OpOb(T_1)} \neg[\![p\cdot\kappa_1\cdot q]\!](\bot)\right)\ \dot\vee\ \left(\bigvee_{q\in OpOb(T_2)} \neg[\![p\cdot\kappa_2\cdot q]\!](\bot)\right),
\end{aligned}$$
where the equation $\overset{*}{=}$ holds because of Lemma 6.3 (i) and (iii):

$$\begin{aligned}
\neg\big(\bot^{p\cdot\kappa_1}\big)\ \dot\vee\ \neg\big(\bot^{p\cdot\kappa_2}\big) &= \Big(\neg\big(\bot^{p\cdot\kappa_1}\big)\vee\neg\big(\bot^{p\cdot\kappa_2}\big)\Big)\wedge\neg\Big(\neg\big(\bot^{p\cdot\kappa_1}\big)\wedge\neg\big(\bot^{p\cdot\kappa_2}\big)\Big)\\
&= \neg\Big(\bot^{p\cdot\kappa_1}\wedge\bot^{p\cdot\kappa_2}\Big)\wedge\Big(\bot^{p\cdot\kappa_1}\vee\bot^{p\cdot\kappa_2}\Big)\\
&= \neg\Big(\big(\bot^{\kappa_1}\big)^p\wedge\big(\bot^{\kappa_2}\big)^p\Big)\wedge\Big(\big(\bot^{\kappa_1}\big)^p\vee\big(\bot^{\kappa_1}\big)^p\Big)\\
&= \neg\Big(\big(\bot^{\kappa_2}\wedge\bot^{\kappa_2}\big)^p\Big)\wedge\Big(\bot^{\kappa_1}\vee\bot^{\kappa_2}\Big)^p\\
&= \neg\big(\bot^p\big)\wedge\top^p\\
&= \neg\big(\bot^p\big).
\end{aligned}$$

Here we use that $(-)^p$ is affine (*i.e.* preserves non-empty joins).

(iv) Basically as in (ii).

(v) The first implication follows because predicate lifting is monotone: $x\in[\![p]\!](\bot)\Rightarrow c(x)\in\bot^p\subseteq\{d\}^p\Rightarrow x\in[\![p]\!](d)$. This holds for every $d\in D$. The second implication follows by applying the inverse image function c^{-1} to the implication of Lemma 6.4 (i). □

The properties in this lemma give rise to a definition.

7.2. Definition. Let T be a polynomial functor. A *T-decomposition* of a set X is a collection of subsets $X_p\subseteq X$, for $p\in OpOb(T)$, satisfying:

(i) X is covered by the X_p:

$$\bigvee_{p\in OpOb(T)} X_p = \top.$$

(ii) For each path $T \overset{p}{\rightsquigarrow} T_1\times T_2$, and $i\in\{1,2\}$,

$$\bigvee_{q\in OpOb(T_1\times T_2)} X_{p\cdot q} = \bigvee_{q\in OpOb(T_i)} X_{p\cdot\pi_i\cdot q}.$$

(iii) For each path $T \overset{p}{\rightsquigarrow} T_1+T_2$,

$$\bigvee_{q\in OpOb(T_1+T_2)} X_{p\cdot q} = \Big(\bigvee_{q\in OpOb(T_1)} X_{p\cdot\kappa_1\cdot q}\Big)\ \dot\vee\ \Big(\bigvee_{q\in OpOb(T_2)} X_{p\cdot\kappa_2\cdot q}\Big)$$

(iv) For each path $T \overset{p}{\rightsquigarrow} T_1^E$ and $e \in E$,

$$\bigvee_{q \in OpOb(T_1^E)} X_{p \cdot q} \quad = \quad \bigvee_{q \in OpOb(T_1)} X_{p \cdot \mathrm{ev}(e) \cdot q}.$$

7.3. Definition. Let T be a polynomial functor with a path $T \overset{p}{\rightsquigarrow} S$. For a coalgebra $c: X \to T(X)$ we can define a function $c_p: \neg[\![p]\!](\perp) \to S(X)$ as:

$$c_p \stackrel{\text{def}}{=} \left(\neg[\![p]\!](\perp) = c^{-1}\Big(\neg(\perp^p)\Big) \xrightarrow{\quad c \quad} \neg(\perp^p) \xrightarrow{\quad \widehat{p} \quad} S(X) \right)$$

where $\widehat{p}$ is introduced in Lemma 6.4 (ii).

In particular what we call the *decomposition* of the coalgebra $c: X \to T(X)$ consists of these functions c_p for p an operator or observer of T. It thus consists of the two collection of functions:

$$\Big(\neg[\![p]\!](\perp) \xrightarrow{c_p} X \Big)_{(T \overset{p}{\rightsquigarrow} \mathrm{Id}) \in Ob(T)} \quad \Big(\neg[\![p]\!](\perp) \xrightarrow{c_p} D \Big)_{(T \overset{p}{\rightsquigarrow} D) \in Op(T)}.$$

Notice that these functions c_p in a decomposition can be considered as partial functions $X \to X$ and $X \to D$. They describe the outcome (if any) of the coalgebra c along the path p, which is either a state or an observation.

The next result describes how functions like in a decomposition can be used to build a coalgebra. How this is done is based on ideas from [Röß00, Röß99].

7.4. Proposition. *Let S be a polynomial functor occurring in a polynomial functor T via a path $T \overset{p}{\rightsquigarrow} S$. Assume a T-decomposition $(X_p \subseteq X)_{p \in OpOb(T)}$ of a set X together with operator- and observer-indexed functions $c_p: X_p \to X$ for $(T \overset{p}{\rightsquigarrow} Id) \in Op(T)$ and $c_p: X_p \to D$ for $(T \overset{p}{\rightsquigarrow} D) \in Ob(T)$. Then:*

(i) *A function $c_p: \left(\bigvee_{q \in OpOb(S)} X_{p \cdot q}\right) \to S(X)$ can be assembled from these data.*

(ii) *In particular, for $p = \langle\rangle$ and thus $S = T$, a coalgebra $c_{\langle\rangle}: X \to T(X)$ results.*

Proof. The last point follows directly from the first one, using Definition 7.2 (i). Therefore we concentrate on the first point. It is proved by induction on S.

- If S is the identity functor, then $OpOb(S) = \{\langle\rangle\}$ and we already have $c_p: X_p \to X = S(X)$.
- Similarly, if S is a constant functor D, then $OpOb(S) = \{\langle\rangle\}$ and we can use $c_p: X_p \to D = S(X)$.
- If $S = S_1 \times S_2$, we define subsets $X_i \subseteq X$ for $i = 1$ and $i = 2$, by:

  $$X_i \stackrel{\text{def}}{=} \bigvee_{q \in OpOb(S_i)} X_{\pi_i \cdot q} \quad = \quad \bigvee_{q \in OpOb(S)} X_q$$

  By induction hypothesis we have $c_{p \cdot \pi_i}: X_i \to S(X)$. Thus we can define $c_p = \langle c_{p \cdot \pi_1}, c_{p \cdot \pi_2} \rangle$.

- If $S = S_1 + S_2$, we define subsets $X_i \subseteq X$, both for $i = 1$ and $i = 2$, by:

$$X_i \quad = \bigvee_{q \in OpOb(S_i)} X_{\kappa_1 \cdot q} \qquad \text{where} \qquad X_1 \dot{\vee} X_2 \quad = \bigvee_{q \in OpOb(S)} X_q$$

  By induction hypothesis, we get two coalgebras $c_{p\cdot\kappa_i}\colon X_i \to S_i(X)$, and so we can define

$$c(x) \quad = \quad \begin{cases} \kappa_1(c_{p\cdot\kappa_1}(x)) & \text{if } x \in X_1 \\ \kappa_2(c_{p\cdot\kappa_2}(x)) & \text{if } x \in X_2. \end{cases}$$

- If $S = S_1^E$, then for each $e \in E$ we define

$$X_e \overset{\text{def}}{=} \bigvee_{q \in OpOb(S_1)} X_{\text{ev}(e)\cdot q} \quad = \bigvee_{q \in OpOb(S)} X_q$$

  By induction hypothesis we have $c_{p\cdot\text{ev}(e)}\colon X_e \to S_1(X)$. Thus we can define $c_p(x)(e) = c_{p\cdot\text{ev}(e)}(x)$. □

The following two lemmas elaborate on this coalgebra assembly.

7.5. Lemma. *Consider a coalgebra $c\colon X \to T(X)$ with its decomposition like in Definition 7.2. The resulting assembly following Proposition 7.4 is c itself.*

Proof. Recall the functions $c_p\colon \neg[\![p]\!](\bot) \to S(X)$ for arbitrary paths $T \overset{p}{\rightsquigarrow} S$ from Definition 7.2, giving rise to the operator- and observer-indexed collection of functions $(c_p)_{p \in OpOb(T)}$. Let us write $c'_p\colon \left(\bigvee_{q \in OpOb(S)} \neg[\![p \cdot q]\!](\bot)\right) \to S(X)$ for the resulting function following Proposition 7.4 (i). One then proves that $c_p = c'_p$, by induction on S. This yields the required result for $p = \langle\rangle$. □

7.6. Lemma. *Suppose we have a pair of sets X, Y with T-decompositions $(X_p \subseteq X)$ and $(Y_p \subseteq Y)$ and functions $c_p\colon X_p \to X$, $d_p\colon Y_p \to Y$, for $T \overset{p}{\rightsquigarrow} Id$, and $c_p\colon X_p \to D$, $d_p\colon Y_p \to D$, for $T \overset{p}{\rightsquigarrow} D$. If a function $f\colon X \to Y$ restricts to $X_p \to Y_p$ and satisfies*

$$\begin{tikzcd} X_p \arrow[r, "f"] \arrow[d, "c_p"'] & Y_p \arrow[d, "d_p"] \\ Y \arrow[r, "f"'] & Y \end{tikzcd} \qquad \begin{tikzcd} X_p \arrow[r, "f"] \arrow[d, "c_p"'] & Y_p \arrow[d, "d_p"] \\ D \arrow[r, equal] & D \end{tikzcd}$$

for each $p \in OpOb(T)$, then f is a homomorphism from $X \overset{c}{\to} T(X)$ to $Y \overset{d}{\to} T(Y)$ between the assembled coalgebras.

Proof. By induction on S in $T \overset{p}{\rightsquigarrow} S$ one shows that f restricts to a function $\left(\bigvee_{q \in OpOb(S)} X_{p\cdot q}\right) \to \left(\bigvee_{q \in OpOb(S)} Y_{p\cdot q}\right)$ and makes the following diagram commute.

$$\begin{tikzcd} \left(\bigvee_{q \in OpOb(S)} X_{p\cdot q}\right) \arrow[r, "f"] \arrow[d, "c_p"'] & \left(\bigvee_{q \in OpOb(S)} Y_{p\cdot q}\right) \arrow[d, "d_p"] \\ S(X) \arrow[r, "S(f)"'] & S(Y) \end{tikzcd}$$

This requires Lemma 6.3 (iv). □

8 Coalgebras and Boolean algebras with operators

In this section we introduce a variant on the notion of Boolean algebra with operators (BAO) from [JT51]. Such algebras form the basic structure in the algebraic semantics of modal logic, see *e.g.* [Lem66, Gol93, BdRV]. Subsequently we show how to obtain a BAO from a coalgebra of a polynomial functor, and vice-versa, resulting in an adjunction.

8.1. Definition. Let T be a polynomial functor all of whose constants are finite. A *Boolean algebra with operators of type* T (a T-BAO, for short) consists of a Boolean algebra B together with functions

$$B \xrightarrow{[\![\,p\,]\!]} B \qquad\qquad D \cup \{\bot\} \xrightarrow{[\![\,q\,]\!]} B$$

for operators $T \overset{p}{\rightsquigarrow} \mathrm{Id}$ and observers $T \overset{q}{\rightsquigarrow} D$. The operators $[\![\,p\,]\!]$ are required to preserve finite meets $\top, \wedge$ and also binary joins $\vee$ (so that they are affine). Additionally, the requirements (i)–(v) from Lemma 7.1 are required to hold.

Given two BAOs B_1, B_2 of type T, a *BAO-morphism* $B_1 \to B_2$ is a function preserving the Boolean algebra structure which commutes with the interpretations of the operators and observers. This yields a category $\mathbf{BAO}(T)$.

The first step in the translations between coalgebras and BAOs is the following. It is based on the constructions and results in the previous two sections.

8.2. Proposition. *Let T be a polynomial functor. There is a functor*

$$\boldsymbol{CoAlg}(T)^{op} \xrightarrow{\quad \mathcal{P} \quad} \boldsymbol{BAO}(T)$$

given on objects by $(X \overset{c}{\to} T(X)) \mapsto \mathcal{P}(X)$ with operators and observers as described in Definition 7.2, and on morphisms by $f \mapsto \mathcal{P}(f) = f^{-1}$.

Proof. The requirements for the observers and operators of a coalgebra obviously hold, and therefore we concentrate on the homomorphism part of the proposition. Assume a coalgebra homomorphism $(X \overset{c}{\to} T(X)) \xrightarrow{f} (Y \overset{d}{\to} T(Y))$. The inverse image function $f^{-1}: \mathcal{P}(Y) \to \mathcal{P}(X)$ preserve the Boolean structure of powersets. It also commutes with predicate lifting, see Lemma 6.3 (iv). Then for an operator $T \overset{p}{\rightsquigarrow} \mathrm{Id}$,

$$\begin{aligned}
(f^{-1} \circ [\![\,p\,]\!]^d)(\alpha) &= \{x \in X \mid d(f(x)) \in \alpha^p\} \\
&= \{x \in X \mid T(f)(c(x)) \in \alpha^p\} \\
&= \{x \in X \mid c(x) \in T(f)^{-1}(\alpha^p)\} \\
&= \{x \in X \mid c(x) \in (f^{-1}(\alpha))^p\} \\
&= ([\![\,p\,]\!]^c \circ f^{-1})(\alpha).
\end{aligned}$$

Similarly for an observer $T \overset{p}{\rightsquigarrow} D$. □

In [Jac99] the temporal logic of coalgebras is expressed in terms of a Galois connection with only two operators, one for nexttime and one for lasttime. This nexttime operator is a "broad" one, which involves all possible steps that can be made by a coalgebra. It is the combination of all the "pathwise" operators that we are considering here. Algebraically, these nexttime and lasttime operators occur in so-called Galois algebras, introduced in [Kar98]. A functor much like the one in the previous propostion is defined in [Jac99], of the form $\mathbf{CoAlg}(T)^{\mathrm{op}} \to \mathbf{GA}$, where $\mathbf{GA}$ is a suitable category of Galois algebras. In this paper we use the category $\mathbf{BAO}(T)$ instead of $\mathbf{GA}$, because it has a richer structure and thus allows us to construct a functor in the opposite direction. This will be done next.

It will be shown how the collection of ultrafilters of a T-BAO carries a T-coalgebra structure, for a polynomial functor T with only finite constant sets. This construction resembles the canonical model construction in [Röß00, Röß99], via Proposition 7.4, but is different in its reliance on the subtle structure of operators and observers as outlined in the previous two sections.

8.3. Proposition. *Let T be a polynomial functor all of whose constant sets are finite. If B is Boolean algebra with operators of type T, then there is a coalgebra structure $\mathsf{clg}(B) \colon \mathsf{spec}\, B \to T(\mathsf{spec}\, B)$.*

This construction yields a functor

$$\boldsymbol{BAO}(T) \xrightarrow{\ \mathsf{spec}\ } \boldsymbol{CoAlg}(T)^{op}$$

which sends a morphism $h \colon B_1 \to B_2$ of T-BAOs to a coalgebra homomorphism $\mathsf{spec}\, h = h^{-1} \colon \mathsf{spec}\, B_2 \to \mathsf{spec}\, B_1$.

Proof. Let B be a T-BAO. We shall use Proposition 7.4 to obtain the coalgebra structure $\mathsf{clg}(B) \colon \mathsf{spec}\, B \to T(\mathsf{spec}\, B)$. What we thus need is a decomposition $\big((\mathsf{spec}\, B)_p \subseteq \mathsf{spec}\, B\big)_{p \in OpOb(T)}$ with appropriate functions. It is defined as follows. For an operator $T \overset{p}{\rightsquigarrow} \mathrm{Id}$ put

$$\big(\mathsf{spec}\, B\big)_p \quad = \quad \{U \in \mathsf{spec}\, B \mid \neg[\![\, p \,]\!](\bot) \in U\}$$

with function $[\![\, p \,]\!]^{-1} \colon (\mathsf{spec}\, B)_p \to \mathsf{spec}\, B$. It is well-defined by Lemma 2.3. Similarly, for $T \overset{p}{\rightsquigarrow} D$ put

$$\big(\mathsf{spec}\, B\big)_p \quad = \quad \{U \in \mathsf{spec}\, B \mid \neg[\![\, p \,]\!](\bot) \in U\}$$

with $(\mathsf{spec}\, B)_p \to D$ defined via the implication $\neg[\![\, p \,]\!](\bot) \supset \dot{\bigvee}_{d \in D} [\![\, p \,]\!](d)$ from Lemma 7.1 (v). Since it is $\top$, it is an element of each $U \in (\mathsf{spec}\, B)_p$, yielding $\dot{\bigvee}_{d \in D} [\![\, p \,]\!](d) \in U$, and thus $[\![\, p \,]\!](d) \in U$, for a unique $d \in D$. This yields the function $(\mathsf{spec}\, B)_p \to D$.

We still have to check that the subsets $(\mathsf{spec}\, B)_p \subseteq \mathsf{spec}\, B$ for $p \in OpOb(T)$ form a T-decomposition. We check two clauses.

$$\begin{aligned}
\bigvee_{p \in OpOb(T)} \big(\mathsf{spec}\, B\big)_p \quad &= \quad \{U \mid \exists p \in OpOb(T).\, \neg[\![\, p \,]\!](\bot) \in U\} \\
&= \quad \{U \mid \textstyle\bigvee_{p \in OpOb(T)} \neg[\![\, p \,]\!](\bot) \in U\} \\
&= \quad \{U \mid \top \in U\} \\
&= \quad \mathsf{spec}\, B.
\end{aligned}$$

And for a path $T \overset{p}{\rightsquigarrow} T_1 + T_2$,

$$
\begin{aligned}
\bigvee_{q \in OpOb(T_1+T_2)} (\mathsf{spec}\, B)_{p \cdot q} &= \{U \mid \bigvee_{q \in OpOb(T_1+T_2)} \neg[\![p \cdot q]\!](\bot) \in U\} \\
&= \{U \mid \ \big(\bigvee_{q_1 \in OpOb(T_1)} \neg[\![p \cdot \kappa_1 \cdot q_1]\!](\bot) \in U\big) \ \dot{\vee} \\
&\qquad\qquad \big(\bigvee_{q_2 \in OpOb(T_2)} \neg[\![p \cdot \kappa_2 \cdot q_2]\!](\bot) \in U\big)\} \\
&= \{U \mid \bigvee_{q_1 \in OpOb(T_1)} \neg[\![p \cdot \kappa_1 \cdot q_1]\!](\bot) \in U\} \ \dot{\vee} \\
&\qquad \{U \mid \bigvee_{q_2 \in OpOb(T_2)} \neg[\![p \cdot \kappa_2 \cdot q_2]\!](\bot) \in U\} \\
&= \left(\bigvee_{q_1 \in OpOb(T_1)} (\mathsf{spec}\, B)_{p \cdot \kappa_1 \cdot q_1} \right) \dot{\vee} \\
&\qquad \left(\bigvee_{q_2 \in OpOb(T_2)} (\mathsf{spec}\, B)_{p \cdot \kappa_1 \cdot q_2} \right).
\end{aligned}
$$

Finally, in order to show that a homomorphism $h: B_1 \to B_2$ of T-BAOs yields a homomorphism of coalgebras $h^{-1}: \mathsf{spec}\, B_2 \to \mathsf{spec}\, B_1$, we use Lemma 7.6. Thus we have to show:

(i) h^{-1} restricts appropriately: $U \in (\mathsf{spec}\, B_2)_p \Rightarrow h^{-1}(U) \in (\mathsf{spec}\, B_1)_p$.

(ii) h^{-1} commutes with the functions $(\mathsf{spec}\, B_i)_p \to \mathsf{spec}\, B_i$ and $(\mathsf{spec}\, B_i)_p \to D$.

As to (i), if $\neg[\![p]\!]^{B_2}(\bot) \in U$, then $\neg[\![p]\!]^{B_1}(\bot) \in h^{-1}(U)$ since $h(\neg[\![p]\!]^{B_1}(\bot)) = \neg h([\![p]\!]^{B_1}(\bot)) = \neg[\![p]\!]^{B_2}(h(\bot)) = \neg[\![p]\!]^{B_2}(\bot) \in U$. Similarly for (ii). $\square$

We leave it as an exercise to the reader to check that the ultrafilter extension of the stack coalgebra described in Subsection 5.2 is obtained by applying the recipe in this proof to the Boolean algebra with operators obtained in Subsection 5.1.

We are now in a position to put the two functors $\mathcal{P}: \mathbf{CoAlg}(T)^{\mathrm{op}} \to \mathbf{BAO}(T)$ from Proposition 8.2 and $\mathsf{spec}: \mathbf{BAO}(T) \to \mathbf{CoAlg}(T)^{\mathrm{op}}$ from Proposition 8.3 together in adjunction $\mathsf{spec} \dashv \mathcal{P}$, using the earlier mentioned unit η and counit ε functions.

8.4. Theorem. *Let T be a polynomial functor all of whose constant sets are finite. Then there is an adjunction $\mathsf{spec} \dashv \mathcal{P}$ between T-BAOs and T-coalgebras in a situation:*

$$
\begin{tikzcd}
\boldsymbol{BAO}(T) \arrow[d, bend right=50, "\mathsf{spec}"'] \arrow[d, phantom, "\dashv"] \\
\boldsymbol{CoAlg}(T)^{op} \arrow[u, bend right=50, "\mathcal{P}"']
\end{tikzcd}
$$

with unit $\eta_B: B \to \mathcal{P}(\mathsf{spec}\, B)$ in $\boldsymbol{BAO}(T)$ and counit $\varepsilon_{c: X \to T(X)}: \mathsf{spec}\,\mathcal{P}(X) \to X$ in $\boldsymbol{CoAlg}(T)^{op}$ given by:

$$
\eta_B(b) = \{U \in \mathsf{spec}\, B \mid b \in U\} \qquad \textit{and} \qquad \varepsilon_{c: X \to T(X)}(x) = \{\alpha \in \mathcal{P}(X) \mid x \in \alpha\}.
$$

Proof. Recall from Proposition 8.3 that we write $\mathsf{clg}(B): \mathsf{spec}\, B \to T(\mathsf{spec}\, B)$ for the coalgebra induced on a T-BAO B. The triangular identities standardly hold:

$$
\begin{tikzcd}
\mathcal{P}(X) \arrow[r, "\eta_{\mathcal{P}(X)}"] \arrow[dr, equal] & \mathcal{P}(\mathsf{spec}\,\mathcal{P}(X)) \arrow[d, "\mathcal{P}(\varepsilon_{c: X \to T(X)})"] \\
& \mathcal{P}(X)
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathsf{spec}\, B \arrow[r, "\mathsf{spec}\, \eta_B"] \arrow[dr, equal] & \mathsf{spec}\,\mathcal{P}(\mathsf{spec}\, B) \arrow[d, "\varepsilon_{\mathsf{clg}(B)}"] \\
& \mathsf{spec}\, B
\end{tikzcd}
$$

where the first one lives in $\mathbf{BAO}(T)$ and the second one in $\mathbf{CoAlg}(T)^{\mathrm{op}}$, and should therefore be read in the reverse direction.

Further, it is standard that the η's are homomorphisms of Boolean algebras. It requires more work to see that they also commute with the modal operators and obervers of T. The result follows (for $p = \langle\rangle$) from the following.

Claim. For each path $T \overset{p}{\rightsquigarrow} S$, put

$$
\begin{array}{rcl}
(\mathsf{spec}\,B)_p & = & \{U \in \mathsf{spec}\,B \mid \exists q \in \mathit{OpOb}(S).\, \neg[\![\, p \cdot q \,]\!](\bot) \in U\} \\
\eta_p(b) & = & \{U \in (\mathsf{spec}\,B)_p \mid b \in U\} \\
[\![\, q \,]\!]_p(\alpha) & = & \{U \in (\mathsf{spec}\,B)_p \mid \mathsf{clg}(B)_p(U) \in \alpha^q\} \\
[\![\, r \,]\!]_p(d) & = & \{U \in (\mathsf{spec}\,B)_p \mid \mathsf{clg}(B)_p(U) \in \{d\}^r\} \\
[\![\, r \,]\!]_p(\bot) & = & \{U \in (\mathsf{spec}\,B)_p \mid \mathsf{clg}(B)_p(U) \in \bot^r\}
\end{array}
$$

where in the last three cases we assume $S \overset{q}{\rightsquigarrow} \mathrm{Id}$, $S \overset{r}{\rightsquigarrow} D$, $\alpha \subseteq \mathsf{spec}\,B$, and $\mathsf{clg}(B)_p \colon (\mathsf{spec}\,B)_p \to S(\mathsf{spec}\,B)$ is the function defined according to Proposition 7.4 (i). Then the following diagrams commute.

$$
\begin{tikzcd}
B \arrow[r, "\eta"] \arrow[d, "{[\![\, p \cdot q \,]\!]}"'] & \mathcal{P}(\mathsf{spec}\,B) \arrow[d, "{[\![\, q \,]\!]_p}"] \\
B \arrow[r, "\eta_p"'] & \mathcal{P}((\mathsf{spec}\,B)_p)
\end{tikzcd}
\qquad
\begin{tikzcd}
D \cup \{\bot\} \arrow[r, equal] \arrow[d, "{[\![\, p \cdot r \,]\!]}"'] & D \cup \{\bot\} \arrow[d, "{[\![\, r \,]\!]_p}"] \\
B \arrow[r, "\eta_p"'] & \mathcal{P}((\mathsf{spec}\,B)_p)
\end{tikzcd}
$$

This is shown by induction on the structure of S. We do two cases, both for operators and for observers.

- If S is the identity functor, then we only have one operator, which satisfies:

$$
\begin{array}{rcl}
[\![\, \langle\rangle \,]\!]_p(\eta(b)) & = & \{U \in (\mathsf{spec}\,B)_p \mid \mathsf{clg}(B)_p(U) \in \eta(b)\} \\
& = & \{U \in (\mathsf{spec}\,B)_p \mid b \in [\![\, p \,]\!]^{-1}(U)\} \\
& = & \{U \in (\mathsf{spec}\,B)_p \mid [\![\, p \,]\!](b) \in U\} \\
& = & \eta_p([\![\, p \,]\!](b)).
\end{array}
$$

- If S is a product $S_1 \times S_2$, we shall do the case of the first projection. For an operator $q \in \mathit{Op}(S_1)$,

$$
\begin{array}{rcl}
[\![\, \pi_1 \cdot q \,]\!]_p(\eta(b)) & = & \{U \in (\mathsf{spec}\,B)_p \mid \mathsf{clg}(B)_p(U) \in \eta(b)^{\pi_1 \cdot q}\} \\
& = & \{U \in (\mathsf{spec}\,B)_p \mid \pi_1(\mathsf{clg}(B)_p(U)) \in \eta(b)^q\} \\
& = & \{U \in (\mathsf{spec}\,B)_p \mid \mathsf{clg}(B)_{p \cdot \pi_1}(U) \in \eta(b)^q\} \\
& \overset{\text{(IH)}}{=} & \{U \in (\mathsf{spec}\,B)_p \mid U \in \eta_{p \cdot \pi_1}([\![\, p \cdot \pi_1 \cdot q \,]\!](b))\} \\
& = & \eta_p([\![\, p \cdot \pi_1 \cdot q \,]\!](b))
\end{array}
$$

Next we consider observers.

- If S is a constant functor D, there is only one observer, with:

$$\begin{array}{rcl}
[\![\, \langle\rangle \,]\!]_p(d) & = & \{U \in (\mathsf{spec}\, B)_p \mid \mathsf{clg}(B)_p(U) \in \{d\}^{\langle\rangle}\} \\
& = & \{U \in (\mathsf{spec}\, B)_p \mid \mathsf{clg}(B)_p(U) = d\} \\
& = & \{U \in (\mathsf{spec}\, B)_p \mid [\![\, p \,]\!](d) \in U\} \\
& = & \eta_p([\![\, p \,]\!](d)). \\
[\![\, \langle\rangle \,]\!]_p(\bot) & = & \{U \in (\mathsf{spec}\, B)_p \mid \mathsf{clg}(B)_p(U) \in \bot^{\langle\rangle}\} \\
& = & \{U \in (\mathsf{spec}\, B)_p \mid \bot\} \\
& = & \{U \in (\mathsf{spec}\, B)_p \mid [\![\, p \,]\!](\bot) \in U\} \\
& = & \eta_p([\![\, p \,]\!](\bot)).
\end{array}$$

- If S is a coproduct $S_1 + S_2$, we have:

$$\begin{array}{ll}
\multicolumn{2}{l}{[\![\, \kappa_1 \cdot r \,]\!]_p(d)} \\
= & \{U \in (\mathsf{spec}\, B)_p \mid \mathsf{clg}(B)_p(U) \in \{d\}^{\kappa_1 \cdot r}\} \\
= & \{U \in (\mathsf{spec}\, B)_p \mid \forall V.\, \mathsf{clg}(B)_p(U) = \kappa_1(V) \Rightarrow V \in \{d\}^r\} \\
= & \{U \in (\mathsf{spec}\, B)_p \mid U \in (\mathsf{spec}\, B)_{p\cdot\kappa_1} \Rightarrow \mathsf{clg}(B)_{p\cdot\kappa_1}(U) \in \{d\}^r\} \\
\stackrel{\text{(IH)}}{=} & \{U \in (\mathsf{spec}\, B)_p \mid U \in (\mathsf{spec}\, B)_{p\cdot\kappa_1} \Rightarrow U \in \eta_{p\cdot\kappa_1}([\![\, p \cdot \kappa_1 \cdot r \,]\!](d))\} \\
\stackrel{*}{=} & \{U \in (\mathsf{spec}\, B)_p \mid U \in \eta_p([\![\, p \cdot \kappa_1 \cdot r \,]\!](d))\} \\
= & \eta_p([\![\, p \cdot \kappa_1 \cdot r \,]\!](d))
\end{array}$$

  The inclusion ($\supseteq$) of the marked equation $\stackrel{*}{=}$ is obvious, and for ($\subseteq$) we reason as follows. If $U \notin (\mathsf{spec}\, B)_{p\cdot\kappa_1}$, then $[\![\, p \cdot \kappa_1 \cdot q \,]\!](\bot) \in U$ for each $q \in OpOb(S_1)$. In particular, $[\![\, p \cdot \kappa_1 \cdot r \,]\!](\bot) \in U$. The first implication in Lemma 7.1 (v) then yields $[\![\, p \cdot \kappa_1 \cdot r \,]\!](d) \in U$.

  The case of $\bot$ instead of $d \in D$ is handled similarly.

We still have to prove that the counits ε are homomorphisms of coalgebras. This follows from Lemmas 7.5 and 7.6. First, $\varepsilon\colon X \to \mathsf{spec}\,\mathcal{P}(X)$ restricts to $\neg[\![\, p \,]\!](\bot) \to \big(\mathsf{spec}\,\mathcal{P}(X)\big)_p$, since

$$\begin{array}{rcl}
x \in \neg[\![\, p \,]\!](\bot) & \Rightarrow & x \notin [\![\, p \,]\!](\bot) \\
& \Rightarrow & [\![\, p \,]\!](\bot) \notin \varepsilon(x) \\
& \Rightarrow & \neg[\![\, p \,]\!](\bot) \in \varepsilon(x) \\
& \Rightarrow & \varepsilon(x) \in \big(\mathsf{spec}\,\mathcal{P}(X)\big)_p.
\end{array}$$

Second, ε makes the following diagrams commute:

$$\begin{tikzcd}
\neg[\![\, p \,]\!](\bot) \arrow[r, "\varepsilon"] \arrow[d, "c_p"'] & \big(\mathsf{spec}\,\mathcal{P}(X)\big)_p \arrow[d, "\mathsf{clg}(\mathcal{P}(X))_p"] \\
X \arrow[r, "\varepsilon"'] & \mathsf{spec}\,\mathcal{P}(X)
\end{tikzcd}
\qquad
\begin{tikzcd}
\neg[\![\, p \,]\!](\bot) \arrow[r, "\varepsilon"] \arrow[d, "c_p"'] & \big(\mathsf{spec}\,\mathcal{P}(X)\big)_p \arrow[d, "\mathsf{clg}(\mathcal{P}(X))_p"] \\
D \arrow[r, equal] & D
\end{tikzcd}$$

By Lemma 6.4 (ii), for an operator $T \overset{p}{\rightsquigarrow} \mathrm{Id}$,

$$
\begin{array}{rcl}
\big(\mathsf{clg}(\mathcal{P}(X))_p \circ \varepsilon\big)(x) & = & [\![\, p \,]\!]^{-1}(\varepsilon(x)) \\
& = & \{\alpha \in \mathcal{P}(X) \mid [\![\, p \,]\!](\alpha) \in \varepsilon(x)\} \\
& = & \{\alpha \in \mathcal{P}(X) \mid c(x) \in \alpha^p\} \\
& = & \{\alpha \in \mathcal{P}(X) \mid \widehat{p}(c(x)) \in \alpha\} \\
& = & \big(\varepsilon \circ c_p\big)(x).
\end{array}
$$

And similarly for an observer $T \overset{p}{\rightsquigarrow} D$,

$$
\begin{array}{rcl}
\big(\mathsf{clg}(\mathcal{P}(X))_p \circ \varepsilon\big)(x) = d & \Leftrightarrow & [\![\, p \,]\!](d) \in \varepsilon(x) \\
& \Leftrightarrow & c(x) \in \{d\}^p \\
& \Leftrightarrow & \widehat{p}(c(x)) \in \{d\} \\
& \Leftrightarrow & c_p(x) = d.
\end{array}
$$

□

9 Conclusions

This paper started from modal operators and observers for polynomial functors in (a logic for) coalgebraic specification, and developed general definitions and results to derive appropriate Boolean algebras with operators from coalgebras of such functors. In the reverse direction it was shown how to obtain coalgebras from these Boolean algebras with operators. This resulted in an adjunction. In the end, when we return to specification, these constructions allow us to do coalgebraic specification without (explicit) coalgebras, using modal operators and obervers only.

Acknowledgements

Thanks to Martin Rößiger and Yde Venema for helpful discussions.

References

[BdRV] P. Blackburn, M. de Rijke, and Y. Venema. Modal logic. Forthcoming book.

[Bor94] F. Borceux. *Handbook of Categorical Algebra*, volume 50, 51 and 52 of *Encyclopedia of Mathematics*. Cambridge Univ. Press, 1994.

[BW90] M. Barr and Ch. Wells. *Category Theory for Computing Science*. Prentice Hall, 1990.

[DP90] B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order*. Math. Textbooks. Cambridge Univ. Press, 1990.

[Gol93] R. Goldblatt. *Mathematics of Modality*. CSLI Lecture Notes 43, Stanford, 1993.

[HHJT98] U. Hensel, M. Huisman, B. Jacobs, and H. Tews. Reasoning about classes in object-oriented languages: Logical models and tools. In Ch. Hankin, editor, *European Symposium on Programming*, number 1381 in Lect. Notes Comp. Sci., pages 105–121. Springer, Berlin, 1998.

[HJ98] C. Hermida and B. Jacobs. Structural induction and coinduction in a fibrational setting. *Inf. & Comp.*, 145:107–152, 1998.

[Jac96] B. Jacobs. Objects and classes, co-algebraically. In B. Freitag, C.B. Jones, C. Lengauer, and H.-J. Schek, editors, *Object-Orientation with Parallelism and Persistence*, pages 83–103. Kluwer Acad. Publ., 1996.

[Jac99] B. Jacobs. The temporal logic of coalgebras via Galois algebras. Techn. Rep. CSI-R9906, Comput. Sci. Inst., Univ. of Nijmegen, 1999.

[Jac00] B. Jacobs. Exercises in coalgebraic specification. To appear in the proceedings of the *Mathematics for Information Technology* Summer School and Workshop on *Algebraic and Coalgebraic Methods in the Mathematics of Program Construction*, 2000.

[Joh82] P.T. Johnstone. *Stone Spaces.* Number 3 in Cambridge Studies in Advanced Mathematics. Cambridge Univ. Press, 1982.

[JR97] B. Jacobs and J. Rutten. A tutorial on (co)algebras and (co)induction. *EATCS Bulletin*, 62:222–259, 1997.

[JT51] B. Jónsson and A. Tarski. Boolean algebras with operators I. *Amer. Journ. Math.*, 73:891–939, 1951.

[Kar98] B. von Karger. Temporal algebra. *Math. Struct. in Comp. Sci.*, 8:277–320, 1998.

[Kur00] A. Kurz. Specifying coalgebras with modal logic. *Theor. Comp. Sci.*, 2000. To appear.

[Lan71] S. Mac Lane. *Categories for the Working Mathematician.* Springer, Berlin, 1971.

[Lem66] E.J. Lemmon. Algebraic semantics for modal logics II. *Journ. Symb. Logic*, 31:191–218, 1966.

[Mos99] L.S. Moss. Coalgebraic logic. *Ann. Pure & Appl. Logic*, 1999. To appear.

[Rei95] H. Reichel. An approach to object semantics based on terminal co-algebras. *Math. Struct. in Comp. Sci.*, 5:129–152, 1995.

[Röß99] M. Rößiger. Coalgebras and modal logic. Techn. Rep. MATH-AL-17-1999, Institut für Algebra, Techn. Univ. Dresden, 1999.

[Röß00] M. Rößiger. From modal logic to terminal coalgebras. *Theor. Comp. Sci.*, 2000. To appear.

[Rut99] J. Rutten. Universal coalgebra: a theory of systems. *Theor. Comp. Sci.*, 1999. To appear.

[Vic89] S. Vickers. *Topology Via Logic.* Number 5 in Tracts in Theor. Comp. Sci. Cambridge Univ. Press, 1989.

[Wal91] R.F.C. Walters. *Categories and Computer Science.* Carslaw Publications, Sydney, 1991. Also available as: Cambridge Computer Science Text 28, 1992.