Theoretical Informatics and Applications
Theoret. Informatics Appl. **35** (2001) 31–59

MANY-SORTED COALGEBRAIC MODAL LOGIC: A MODEL-THEORETIC STUDY

Bart Jacobs[1]

Abstract. This paper gives a semantical underpinning for a many-sorted modal logic associated with certain dynamical systems, like transition systems, automata or classes in object-oriented languages. These systems will be described as coalgebras of so-called polynomial functors, built up from constants and identities, using products, coproducts and powersets. The semantical account involves Boolean algebras with operators indexed by polynomial functors, called MBAOs, for Many-sorted Boolean Algebras with Operators, combining standard (categorical) models of modal logic and of many-sorted predicate logic. In this setting we will see Lindenbaum MBAO models as initial objects, and canonical coalgebraic models of maximally consistent sets of formulas as final objects. They will be used to (re)prove completeness results, and Hennessey–Milner style characterisation results for the modal logic, first established by Rößiger.

Mathematics Subject Classification. 03G05, 03G30, 06E25.

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 [13, 21, 26]. Modal logic is a logic for dynamical systems. The connections between the areas of coalgebra and modal logic [1, 4, 5, 11, 17, 20, 22–24] form currently an area of active research.

Keywords and phrases: Modal logic, coalgebra, Boolean algebra with operators.

[1] Department of Computer Science, University of Nijmegen, P.O. Box 9010, 6500 GL Nijmegen, The Netherlands; e-mail: bart@cs.kun.nl URL: http://www.cs.kun.nl/~bart

The following developments constitute the background of the current work.

1. The idea that the functor of a coalgebra determines a certain modal logic was first put forward by Moss [20]. He developed it for very general functors, but the idea was applied by others (Rößiger, Kurz, Jacobs, Goldblatt) mostly to a restricted class of inductively defined "polynomial functors".
2. The idea to extract coalgebraic structure from maximally consistent sets of formulas is due to Rößiger [23,24], and was used by him to prove a completeness result *via* an extension of what is called a canonical model construction in modal logic.
3. The idea to use a *many-sorted* modal logic for coalgebras is due to Venema[2], as used in [30], and was elaborated in [23].
4. The idea to define appropriate Boolean Algebras with Operators (BAOs) for (single-sorted) coalgebraic modal logic, with suitable back-and-forth translations between these algebraic models and coalgebraic (dynamical) models, comes from [12]. There, however, this is elaborated only for polynomial functors without powerset.

The single-sorted approach of [12] involves non-trivial properties of so-called observer and operator paths of maximal length, with a constant or identity functor as codomain. These "global" properties are not so easy to formulate. In the many-sorted approach these properties are replaced by relatively simple "local" requirements about the single steps in paths. This makes the many-sorted approach more convenient, and makes it more suited to handle non-determinism (*via* the powerset functor).

This paper applies the semantical approach from [12] to the many-sorted modal logic from [23]. This involves the move from single-sorted to many-sorted BAOs, *via* the introduction of appropriately indexed BAOs. This follows general ideas in categorical logic (see [10]) where, for example, models of many-sorted predicate logic are described as Boolean algebras indexed by the sorts. Technically, this indexing takes the form of an indexed category or, alternatively, of a fibration. Here we shall use the slightly more elementary notion of indexed category, which, in general, is a functor of the form $\Phi\colon \mathbb{B}^{\mathrm{op}} \to \mathbf{Cat}$, for a base category $\mathbb{B}$ of sorts (where $\mathbf{Cat}$ is the category of categories). For each sort $S \in \mathbb{B}$, the so-called fibre category $\Phi(S)$ will in our case be a Boolean algebra, so that Φ restricts to a functor of the form $\mathbb{B}^{\mathrm{op}} \to \mathbf{BA}_{\wedge}$, where $\mathbf{BA}_{\wedge}$ is the category of Boolean algebras and finite meet $(\top, \wedge)$ preserving functions between them. Note that these maps need not preserve all the Boolean algebra structure. A Lindenbaum construction will give rise to such an indexed BAO, which turns out to be an inital object (Prop. 4.8). Another basic result will be that each coalgebra gives rise to such an indexed BAO, incorporating its logic. This will give a (functorial) translation from dynamic to algebraic models.

The completeness result of [23] (but also [24]) involves a "canonical model" construction of a coalgebra out of maximally consistent sets of formulas. In our setting this construction will be generalized (like in [12]) by formulating it in terms

[2]Expressed in conversation to Rößiger and the present author.

of ultrafilters, and showing that it gives rise to a functor from indexed BAOs to coalgebras, yielding a translation from algebraic to dynamic models. We single out the crucial step in this construction (see Def. 5.1), and show how it can be used to give an alternative translation from indexed BAOs to coalgebras. The latter translation, when applied to the Lindenbaum model, will give rise to a final coalgebra (Th. 5.8). This coalgebra can then be used directly to give a Hennessey–Milner characterisation result [6]: that two elements of coalgebras are bisimilar if and only if they satisfy the same formulas (on states), see also [17, 23, 24].

This paper is organised as follows. It starts with a preliminary section introducing some background information on polynomial functors, on paths between them and predicate lifting, on bisimulations and bisimilarities, on ultrafilters, and on algebraic and dynamical models of standard modal logic. Then, Section 3 introduces our reformulation of the many-sorted coalgebraic modal logic used in [23] (using weak instead of strong nexttime operators), together with the interpretation of this logic in coalgebras. Section 4 introduces our notion of "many-sorted Boolean algebra with operators" (MBAO), as a suitable indexed collection of Boolean algebras with operators. A sound interpretation of the logic is given in such MBAOs, and its completeness is proved *via* a Lindenbaum construction. The latter gives, as usual, an initial object. Also, a functorial translation from coalgebras to MBAOs is given. Section 5 focusses on a translation in the reverse direction. Actually, besides an algebraic analogue of Rößiger's construction we shall introduce another translation which corresponds to what is done in standard modal logic and works better in the sense that it gives rise to an ultrafilter extension result, and to a final coalgebra.

2. Preliminaries

Let **Sets** be the category of sets and functions. We shall be using a particular collection of functors $T\colon \mathbf{Sets} \to \mathbf{Sets}$, as interfaces of coalgebras. These so-called *Kripke polynomial functors* are built up inductively from the identity and constants, using products, coproducts, exponents (with constants) and powersets. Products of sets X, Y will be written as $X \times Y$, with projection functions $X \xleftarrow{\pi_1} X \times Y \xrightarrow{\pi_2} Y$. The set 1 is a singleton set, typically written as $1 = \{*\}$. It can be regarded as the empty product. Coproducts (or disjoint unions, or sums) are denoted by $X+Y$, where, for example $X+Y = \{(x,0) \mid x \in X\} \cup \{(y,1) \mid y \in Y\}$. They come with coprojection functions $X \xrightarrow{\kappa_1} X+Y \xleftarrow{\kappa_2} Y$. The coprojections are injective, disjoint (in the sense that $\kappa_1(x) \neq \kappa_2(y)$, for all $x \in X$ and $y \in Y$), and cover $X+Y$ (*i.e.* each $z \in X+Y$ is either in the image of κ_1 or of κ_2). The collection of functions from a set X to Y is denoted by Y^X. For a function $f\colon Y \to Z$ there is an associated function $f^X\colon Y^X \to Z^X$ by $g \mapsto f \circ g$. The (covariant) powerset functor $\mathcal{P}\colon \mathbf{Sets} \to \mathbf{Sets}$ sends a set Y to the set of its subsets $\mathcal{P}(Y) = \{b \mid b \subseteq Y\}$, and a function $f\colon Y \to Z$ to the function $\mathcal{P}(f)\colon \mathcal{P}(Y) \to \mathcal{P}(Z)$ given by direct image: $b \mapsto f(b) = \{f(y) \mid y \in b\}$.

Definition 2.1. The collection of *Kripke polynomial functors* (KPFs) **Sets** $\to$ **Sets** that we use arises as the least collection satisfying:

1. the identity functor $\mathrm{Id}\colon \mathbf{Sets} \to \mathbf{Sets}$ is a KPF;
2. for each non-empty finite set D, the constant functor $D\colon \mathbf{Sets} \to \mathbf{Sets}$, given by $X \mapsto D$ and $(f\colon X \to Y) \mapsto \mathrm{id}_D$, is a KPF;
3. the product $X \mapsto T_1(X) \times T_2(X)$ of two KPFs T_1, T_2 is a KPF;
4. the coproduct $X \mapsto T_1(X) + T_2(X)$ of two KPFs T_1, T_2 is also a KPF;
5. for a KPF T, and an arbitrary set D the exponent functor $X \mapsto T(X)^D$ is also a KPF;
6. for a KPF T, the functor $X \mapsto \mathcal{P}(T(X))$ is a KPF.

The collection of *finite* KPFs is constructed in the same way, except that in the last point the finite powerset $\mathcal{P}_{\mathrm{fin}}$ is used-instead of the ordinary one.

Finiteness of KPFs will only play a role in the last part of the paper (Sects. 5.2 and 5.3). All earlier results hold for all KPFs. The restriction to non-empty finite sets D in the second point is essential is several places (such as Defs. 3.2, 4.1(3), and 5.1) in order to form finite disjunctions, and use these in relation to ultrafilters. We shall make this explicit, whenever appropriate. This finiteness restriction means that examples can only involve finite sets of data.

A *coalgebra* of a (Kripke polynomial) functor $T\colon \mathbf{Sets} \to \mathbf{Sets}$ consists of a set X, usually called the state space or set of states, together with a function $c\colon X \to T(X)$, giving the operations of the coalgebra. A *(homo)morphism of coalgebras* from $X \xrightarrow{c} T(X)$ to $Y \xrightarrow{d} T(Y)$ is a function $f\colon X \to Y$ 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.

Example 2.2. We briefly mention several examples of coalgebras.

(1) **Kripke structures and labeled transition systems.** A *Kripke structure* or a *frame* consists of a set of "states" X together with a binary "transition" relation $\to$ on X. If $x \to x'$, then x' is a successor state of x. Notice that a state may have multiple successor states (non-determinism). A function $f\colon X \to Y$ between the state spaces of two Kripke structures $(X, \to_X)$ and $(Y, \to_Y)$ is called a *bounded morphism* if it satisfies:

1. $x \to_X x' \Longrightarrow f(x) \to_Y f(x')$;
2. $f(x) \to_Y y \Longrightarrow \exists x' \in X.\, x \to_X x'$ and $f(x') = y$.

It is not hard to see that a Kripke structure $(X, \to)$ corresponds to a coalgebra $X \to \mathcal{P}(X)$, given by $x \mapsto \{x' \in X \mid x \to x'\}$. And also that bounded morphisms correspond to homomorphisms of coalgebras. Thus the category of Kripke structures may be identified with the category $\mathbf{CoAlg}(\mathcal{P})$ of coalgebras of the powerset functor.

A *labeled transition system* (LTS) is like a Kripke structure but has labels in the transition relation. If A is a finite set of labels, then an A-LTS consists of a set of states X with a transition relation $\to \subseteq X \times A \times X$. Equivalently, it is a coalgebra of the KPF $\mathcal{P}(A \times \mathrm{Id})$.

(2) **Bounded stacks.** In object-oriented programming a class is a basic entity combining data and associated operations. These data can be described *via* "attributes" $X \to D$ on the state space X of the class, where D is a non-empty finite set of observable data elements. The associated operations, often called methods in this setting, can be described as acting on (or modifying) the state space. The effect of these methods can then become visible *via* the attributes. Because classes form a state-based notion (unlike data structures), with their operations (attributes plus methods) acting on states, makes a representation *via* coalgebras most natural.

For instance, the attributes and methods of a "bounded stack" (used in [12]) can be described as:

$$\begin{aligned} \mathsf{size}\colon X &\longrightarrow \{0, 1, \ldots, N\} \\ \mathsf{push}\colon X \times D &\longrightarrow X + X \\ \mathsf{pop}\colon X &\longrightarrow X + (D \times X). \end{aligned}$$

These separate operations can equivalently be described *via* a single map, forming a coalgebra of a KPF:

$$X \xrightarrow{\langle \mathsf{size}, \mathsf{push}, \mathsf{pop} \rangle} \{0, 1, \ldots, N\} \times (X + X)^D \times (X + (D \times X)).$$

In this description the types of the operations capture the different possible outcomes: the push operation of adding a data element from D to the stack may fail or succeed, depending on whether the stack is full or not. Coalgebraically, this is reflected in the result type $X + X$. Similarly, the pop operation for removing an element may fail or succeed, depending on whether the stack is empty or not. In the latter case, the pop operation produces an element (in D) together with a (modified) state.

More information on the coalgebraic description of classes in object-oriented languages may be found in [7, 9, 11, 14, 21, 25].

(3) **Deterministic and non-deterministic automata.** Let A be an arbitrary set, often called an alphabet of symbols in this context. A deterministic automaton consists of a set of states X with a transition function $\delta\colon X \times A \to X$ and a subset $F \subseteq X$ of final (or halting) states. This function and subset can be combined into a coalgebra $X \to X^A \times \{0, 1\}$ of the KPF $\mathrm{Id}^A \times \{0, 1\}$. Notice that we ignore initial states, because usually in coalgebra they are considered as extra structure.

Non-deterministic automata have a transition function $\delta\colon X \to \mathcal{P}(X)^A$ that can produce multiple successor states for a single symbol. They correspond to coalgebras of the functor $(\mathcal{P}(\mathrm{Id}))^A \times \{0, 1\}$.

What we see is that the functor describes the kind of computation that can be performed by a coalgebra. And associated with this kind of computations there are appropriate logical operators. This key idea comes from [20]. Making this explicit for Kripke polynomial functors requires a further structural analysis.

2.1. Paths as sorts

Let T be a KPF which contains another KPF S as ingredient, 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, $\mathcal{P}$, $\mathrm{ev}(d)$, for elements $d \in D$ of sets D occurring as exponent in T. Such a path tells us how to find S in T. Note that paths enable us to distinguish different occurrences (if any) of S in T.

Definition 2.3. Let T and S be KPFs. 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.
- $T_1 \times T_2 \overset{\pi_1 \cdot p}{\rightsquigarrow} S$ if $T_1 \overset{p}{\rightsquigarrow} S$, and $T_1 \times T_2 \overset{\pi_2 \cdot p}{\rightsquigarrow} S$ if $T_2 \overset{p}{\rightsquigarrow} S$.
- $T_1 + T_2 \overset{\kappa_1 \cdot p}{\rightsquigarrow} S$ if $T_1 \overset{p}{\rightsquigarrow} S$, and $T_1 + T_2 \overset{\kappa_2 \cdot p}{\rightsquigarrow} S$ if $T_2 \overset{p}{\rightsquigarrow} S$.
- $T^D \overset{\mathrm{ev}(d) \cdot p}{\rightsquigarrow} S$ for all $d \in D$, if $T \overset{p}{\rightsquigarrow} S$.
- $\mathcal{P}(T) \overset{\mathcal{P} \cdot p}{\rightsquigarrow} S$ if $T \overset{p}{\rightsquigarrow} S$.

We shall write $\mathsf{Ing}(T)$ for the set of "ingredient" functors[3] that are used in the inductive construction of T. More precisely, $S \in \mathsf{Ing}(T)$ if and only if there is a path $T \rightsquigarrow S$.

Notice that almost all KPFs, except constant ones, have the identity functor Id as ingredient.

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 $T_1 \overset{p \cdot q}{\rightsquigarrow} T_3$. This leads to a category.

Definition 2.4. We shall write **KPF** for the category with Kripke polynomial functors as objects and paths between them as morphisms. The empty paths are identity morphisms, and composition of paths yields composition in **KPF**.

For a KPF T, we shall write $\mathbf{Ing}(T)$ for the full subcategory of **KPF** with ingredients of T as objects, *i.e.* with objects from $\mathsf{Ing}(T)$.

The following basic constructions will be important later. They involve "predicate lifting" for paths (from [8, 11, 12]).

Definition 2.5. For a path $T \overset{p}{\rightsquigarrow} S$ and an arbitrary set X there is a "predicate lifting" function

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

[3] These are called subfunctors in [23] but this terminology may be confusing, since these ingredients of T have nothing to do with subobject of T in a functor category, as the name subfunctor suggests.

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

- $\alpha^{\langle\rangle} = \alpha$;
- $\alpha^{\pi_1 \cdot p} = \{z \mid \pi_1(z) \in \alpha^p\}$;
  $\alpha^{\pi_2 \cdot p} = \{z \mid \pi_2(z) \in \alpha^p\}$;
- $\alpha^{\kappa_1 \cdot p} = \{z \mid \forall y.\, z = \kappa_1(y) \Rightarrow y \in \alpha^p\}$;
  $\alpha^{\kappa_2 \cdot p} = \{z \mid \forall y.\, z = \kappa_2(y) \Rightarrow y \in \alpha^p\}$;
- $\alpha^{\mathrm{ev}(a) \cdot p} = \{f \mid f(a) \in \alpha^p\}$;
- $\alpha^{\mathcal{P} \cdot p} = \{\beta \mid \beta \subseteq \alpha^p\}$.

The next result from [12] gives some elementary properties of predicate lifting. Proofs are by induction on paths.

Lemma 2.6. 1. *The predicate lifting function $(-)^p \colon \mathcal{P}(S(X)) \to \mathcal{P}(T(X))$ associated with a path $T \overset{p}{\rightsquigarrow} S$ preserves arbitrary meets $\bigwedge$. In the special cases where p is a projection π_i or an evaluation step $ev(d)$, all the Boolean structure is preserved.*

2. *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.*

$$\begin{array}{ccc} \mathcal{P}(T_3(X)) & \xrightarrow{(-)^q} & \mathcal{P}(T_2(X)) \\ & \searrow^{(-)^{p \cdot q}} & \downarrow{\scriptstyle (-)^p} \\ & & \mathcal{P}(T_1(X)) \end{array}$$

3. *Predicate lifting is natural: for an arbitrary function $f \colon X \to Y$ and path $T \overset{p}{\rightsquigarrow} S$ the following diagram commutes.*

$$\begin{array}{ccc} \mathcal{P}(S(X)) & \xrightarrow{(-)^p} & \mathcal{P}(T(X)) \\ {\scriptstyle S(f)^{-1}}\uparrow & & \uparrow{\scriptstyle T(f)^{-1}} \\ \mathcal{P}(S(Y)) & \xrightarrow[(-)^p]{} & \mathcal{P}(T(Y)) \end{array}$$

The last two points allow us to set up appropriately indexed structures *via* predicate lifting. They will be investigated further in Section 4.

Proposition 2.7. *Let $\boldsymbol{BA}_\wedge$ be the category of Boolean algebras and finite meet preserving maps between them. Each KPF T and set X gives rise to a functor*

$$\boldsymbol{Ing}(T)^{\mathrm{op}} \longrightarrow \boldsymbol{BA}_\wedge$$

by:

$$\begin{array}{rcl} S & \longmapsto & \mathcal{P}(S(X)) \\ \big(S_1 \overset{p}{\rightsquigarrow} S_2\big) & \longmapsto & (-)^p \colon \mathcal{P}(S_2(X)) \to \mathcal{P}(S_1(X)). \end{array}$$

This gives an example of what is called an "indexed Boolean algebra", because the functor describes an $\mathsf{Ing}(T)$-indexed collection $\big(\mathcal{P}(S(X))\big)_S$ of Boolean algebras, with appropriate homomorphisms between them.

Actually, the functor T does not really play a role in the previous result — we could write **KPF** instead of $\mathbf{Ing}(T)$ — but we shall use functors as above with some more structure related to T below. Therefore we already use this formulation here.

2.2. Bisimulations and bisimilarity

This subsection recalls the definition of bisimulations and bisimilarity *via* relation lifting, following [8].

Definition 2.8. Let T be a KPF, and let X, Y be arbitrary sets with a relation $R \subseteq X \times Y$ between them.

1. The "lifted" relation $R^T \subseteq T(X) \times T(Y)$ is defined by induction on the structure of the functor T:

$$\begin{array}{rcl} R^A & = & =_A \\ R^{\mathrm{Id}} & = & R \\ R^{T_1 \times T_2} & = & \{(u,v) \mid R^{T_1}(\pi_1(u), \pi_1(v)) \text{ and } R^{T_2}(\pi_2(u), \pi_2(v))\} \\ R^{T_1 + T_2} & = & \{(u,v) \mid \exists x, y.\, u = \kappa_1(x) \text{ and } v = \kappa_1(y) \text{ and } R^{T_1}(x,y) \\ & & \qquad \text{or} \\ & & \qquad \exists x, y.\, u = \kappa_2(x) \text{ and } v = \kappa_2(y) \text{ and } R^{T_2}(x,y)\} \\ R^{T^D} & = & \{(f,g) \mid \forall d \in D.\, R^T(f(d), g(d))\} \\ R^{\mathcal{P}T} & = & \{(\alpha,\beta) \mid \forall x \in \alpha.\, \exists y \in \beta.\, R^T(x,y) \\ & & \qquad \text{and} \\ & & \qquad \forall y \in \beta.\, \exists x \in \alpha.\, R^T(x,y)\}. \end{array}$$

2. Assume now two coalgebras $c \colon X \to T(X)$ and $d \colon Y \to T(Y)$. A relation $R \subseteq X \times Y$ is called a **bisimulation** (with respect to c, d) if, for all $x \in X$ and $y \in Y$,

$$R(x,y) \Longrightarrow R^T(c(x), c(y)).$$

This means that R is closed with respect to the operations c and d.

3. The **bisimilarity** relation $\underline{\leftrightarrows} \subseteq X \times Y$ with respect to coalgebras c and d as before, is defined as the greatest bisimulation (with respect to c, d), *i.e.* as:

$$x \underline{\leftrightarrows} y \iff \exists R \subseteq X \times Y.\, R \text{ is a bisimulation and } R(x, y).$$

Bisimilar states are observationally indistinguishable.

It is not hard to see that bisimilarity $\underline{\leftrightarrows} \subseteq X \times X$ with respect to a single coalgebra $c\colon X \to T(X)$ is an equivalence relation. The following is a standard result, see *e.g.* [26,29] — where it occurs with respect to a different, but equivalent, definition of bisimulation.

Proposition 2.9. *Let T be a KPF which happens to have a final coalgebra $z\colon Z \to T(Z)$. For arbitrary coalgebras $c\colon X \to T(X)$ and $d\colon Y \to T(Y)$, let $!_c : X \to Z$ and $!_d : Y \to Z$ be the corresponding unique coalgebra homomorphisms to the final coalgebra. Then, for all $x \in X$ and $y \in Y$,*

$$x \underline{\leftrightarrows} y \iff !_c(x) = !_d(y).$$

2.3. Ultrafilters

This subsection reviews the basics of ultrafilters, going back to Marshall Stone's work from the thirties [27, 28]. See *e.g.* [2, 15] for modern introductions. Let B be a Boolean algebra, *i.e.* a poset with finite meets $\top, \wedge$ and joins $\bot, \vee$ and a complement operator $\neg$. A typical example is the powerset $\mathcal{P}(A)$ of subsets (also called predicates) of an arbitrary set A. A *filter* of B is a subset $U \subseteq B$ 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 $y \geq x \in U \Rightarrow y \in U$. The least filter containing an arbitrary subset $S \subseteq B$ is the set $\uparrow\!\!S = \uparrow\{\bigwedge \alpha \mid \alpha \subseteq S \text{ finite}\}$—where $\uparrow(-)$ is the upward closure operation: $\uparrow S = \{x \mid \exists y \in S.\, x \geq y\}$.

An *ultrafilter* of B (or also called a maximal or prime filter) is a filter $U \subseteq B$ which satisfies: for each $x \in B$, either $x \in U$ or $\neg x \in U$, but not both. As a consequence, $\bot \notin U$. Also, if $x \vee y \in U$, then either $x \in U$ or $y \in U$. Further, if $\dot\vee$ is the exclusive disjunction $x \mathbin{\dot\vee} y = (x \vee y) \wedge \neg(x \wedge y)$, then $x \mathbin{\dot\vee} y \in U$ implies either $x \in U$ or $y \in U$ but not both. We write $\mathsf{spec}\, B$ for the set of ultrafilters of the Boolean algebra B, and call it the *spectrum* of B. Ultrafilters in a Boolean algebra are the algebraic counterparts of maximally consistent theories in logic.

The following result is often useful. Its proof depends on the Axiom of Choice (in the form of Zorn's lemma), see [15] (I, 2.3 and 2.4) or [2] (Th. 9.13).

Lemma 2.10 (Ultrafilter lemma)**.** *Let F be a filter of a Boolean algebra B, with $\bot \notin F$. Then there is an ultrafilter U of B with $F \subseteq U$.*

The next result relates filters and ultrafilter. It is used for instance in [19] (see Sect. III, Lems. 2 and 3). It resembles the "Scott open filter lemma", see *e.g.* [31] (Lem. 8.2.2).

Lemma 2.11. *Let B be Boolean algebra with a filter $F \subseteq B$. Then*

$$F = \bigcap \{U \in \mathit{spec}\, B \mid F \subseteq U\}.$$

Proof. The direction $(\subseteq)$ is immediate. For the reverse inclusion $(\supseteq)$ assume $\forall U \in \mathsf{spec}\, B.\, F \subseteq U \Rightarrow x \in U$ and $x \notin F$. We first show that $\bot$ is then not in the filter $\uparrow\!(F \cup \{\neg x\})$, because it implies $x \in F$. If $\bot \geq \bigwedge \alpha$ 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 $\bot \in F$ and thus certainly $x \in F$. If $\neg x \in \alpha$, write $\alpha = \{\neg x\} \cup \beta$ with $\beta \subseteq F$ and $\neg x \wedge \bigwedge \beta = \bot$. Then $\bigwedge \beta \leq x$, and thus $x \in F$.

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

Corollary 2.12. *For an arbitrary element a of a Boolean algebra B,*

$$a = \top \iff \forall U \in \mathit{spec}\, B.\, a \in U.$$

Proof. Because, by the previous lemma,

$$\begin{aligned} a = \top &\iff a \in \{\top\} = \bigcap \{U \in \mathsf{spec}\, B \mid \{\top\} \subseteq U\} \\ &\iff \forall U \in \mathsf{spec}\, B.\, a \in U. \end{aligned}$$

□

2.4. Kripke structures and Boolean algebras with operators

This section recalls some standard results [3, 19] about the relation between dynamic models (Kripke structures, see Ex. 2.2(1)) and algebraic models (Boolean algebras with operators, from [16]) of modal logic.

A Kripke structure $c\colon X \to \mathcal{P}(X)$ induces a modal operator $\Box_c\colon \mathcal{P}(X) \to \mathcal{P}(X)$ on the Boolean algebra of predicates on the state space X. It is given by

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

It is not hard to see that $\Box_c$ preserves (arbitrary) meets. It thus forms a *Boolean algebra with operators*.

In the reverse direction, a Boolean algebra B with a unary modal operator $\Box\colon B \to B$ preserving finite meets can be turned into a Kripke structure, with state space $\mathsf{spec}\, B$: a coalgebra structure $c_B\colon \mathsf{spec}\, B \to \mathcal{P}(\mathsf{spec}\, B)$ can be defined as

$$c_B(U) = \{V \in \mathsf{spec}\, B \mid \Box^{-1}(U) \subseteq V\} \qquad \textit{i.e.} \qquad U \to V \iff \Box^{-1}(U) \subseteq V.$$

These translations back-and-forth can both be made functorial. Further, there are canonical comparison maps: a "unit" map $\eta_B\colon B \to \mathcal{P}(\mathsf{spec}\, B)$ given by $b \mapsto$

$\{U \mid b \in U\}$ is a "homomorphisms of BAOs". And for a coalgebra $c\colon X \to \mathcal{P}(X)$, there is the canonical embedding $\varepsilon\colon X \to \mathsf{spec}\,\mathcal{P}(X)$ into the "ultrafilter extension", given by $x \mapsto \{\alpha \mid x \in \alpha\}$. It is folklore knowledge that ε is in general not a morphism of coalgebras (or Kripke structures), but in case each set $c(x)$ is *finite* (when c is "image finite", and the level of non-determinism is limited), it is. We shall see a similar situation in Section 5.2 below.

3. Formulas and rules of many-sorted coalgebraic logic

This section introduces the sort-indexed formulas of many-sorted modal logic, together with their axioms and rules, following [23]. The formulas will be interpreted later as predicates in a coalgebra, see Definition 3.4, and also as elements of Boolean algebras, see Definition 4.2.

Definition 3.1. The way we introduce formulas is to define first what are sometimes called raw formulas, and then to single out the well-typed ones *via* appropriate rules. The raw formulas are:

$$\varphi := \bot \mid \varphi \to \varphi \mid a \mid \mathsf{next}\varphi \mid [\pi_1]\varphi \mid [\pi_2]\varphi \mid [\kappa_1]\varphi \mid [\kappa_2]\varphi \mid [\mathrm{ev}(d)]\varphi \mid [\mathcal{P}]\varphi.$$

For each KPF T we form an indexed collection $\big(\mathsf{Form}_S\big)_{S\in\mathsf{Ing}(T)}$ of subsets of formulas as follows. For each sort $S \in \mathsf{Ing}(T)$ the set Form_S contains *falsum* $\bot$ and is closed under implication $\to$:

$$\frac{}{\bot \in \mathsf{Form}_S} \qquad \frac{\varphi_1 \in \mathsf{Form}_S \quad \varphi_2 \in \mathsf{Form}_S}{\varphi_1 \to \varphi_2 \in \mathsf{Form}_S}.$$

Elements of constants are formulas:

$$\frac{}{a \in \mathsf{Form}_A}\ (a \in A).$$

Further, there are the following closure rules.

$$\frac{\varphi \in \mathsf{Form}_{S_i}}{[\pi_i]\varphi \in \mathsf{Form}_{S_1\times S_2}} \quad \frac{\varphi \in \mathsf{Form}_{S_i}}{[\kappa_i]\varphi \in \mathsf{Form}_{S_1+S_2}} \quad \frac{\varphi \in \mathsf{Form}_S}{[\mathrm{ev}(d)]\varphi \in \mathsf{Form}_{S^D}} \quad \frac{\varphi \in \mathsf{Form}_S}{[\mathcal{P}]\varphi \in \mathsf{Form}_{\mathcal{P}(S)}}.$$

And, if $\mathrm{Id} \in \mathsf{Ing}(T)$,

$$\frac{\varphi \in \mathsf{Form}_T}{\mathsf{next}\varphi \in \mathsf{Form}_{\mathrm{Id}}}.$$

The formulas of sort Id are most interesting, and will be called *state formulas*.

We shall use standard abbreviations: $\neg\varphi = \varphi \to \bot$, $\varphi \vee \psi = \neg\varphi \to \psi$, $\varphi \wedge \psi = \neg(\varphi \to \neg\psi)$, $\varphi \leftrightarrow \psi = (\varphi \to \psi) \wedge (\psi \to \varphi)$, $\varphi \dot\vee \psi = (\varphi \vee \psi) \wedge \neg(\varphi \wedge \psi)$, and also finite generalisations of $\vee$, $\wedge$ and $\dot\vee$.

The next step is to turn these sort-indexed sets of formulas into a deduction calculus. Therefore we introduce axioms and rules.

Definition 3.2. Let T be a KPF. For each ingredient $S \in \mathsf{Ing}(T)$ of T we define the subset $\vdash_S \subseteq \mathsf{Form}_S$ of derivable formulas as the least subset satisfying the following axioms and rules.

For each ingredient S, each Boolean tautology $\varphi \in \mathsf{Form}_S$ satisfies $\vdash_S \varphi$.

Also, the *modus ponens* (MP) rule holds for each ingredient S:

$$\frac{\vdash_S \varphi \to \psi \qquad \vdash_S \varphi}{\vdash_S \psi}.$$

Additionally there are requirements for specific ingredients. For a constant functor A—using that the set A is finite:

$$\vdash_A \dot{\bigvee}_{a \in A} a \quad \text{(Det)}.$$

For the identity functor, if in $\mathsf{Ing}(T)$:

$$\begin{array}{rll} \vdash_{\mathrm{Id}} \mathsf{next}\varphi \leftrightarrow \neg\mathsf{next}\neg\varphi & \text{(Det)} & \qquad \dfrac{\vdash_T \varphi}{\vdash_{\mathrm{Id}} \mathsf{next}\varphi}\ \text{(N)}. \\ \vdash_{\mathrm{Id}} \mathsf{next}(\varphi \to \psi) \to (\mathsf{next}\varphi \to \mathsf{next}\psi) & \text{(K)} & \end{array}$$

For a product functor:

$$\begin{array}{rll} \vdash_{S_1 \times S_2} [\pi_i]\varphi \leftrightarrow \neg[\pi_i]\neg\varphi & \text{(Det)} & \qquad \dfrac{\vdash_{S_i} \varphi}{\vdash_{S_1 \times S_2} [\pi_i]\varphi}\ \text{(N)}. \\ \vdash_{S_1 \times S_2} [\pi_i](\varphi \to \psi) \to ([\pi_i]\varphi \to [\pi_i]\psi) & \text{(K)} & \end{array}$$

For a coproduct functor:

$$\begin{array}{rll} \vdash_{S_1 + S_2} (\neg[\kappa_1]\bot) \mathbin{\dot{\vee}} (\neg[\kappa_2]\bot) & \text{(DC)} & \\ \vdash_{S_1 + S_2} (\neg[\kappa_i]\bot) \to ([\kappa_i]\varphi \leftrightarrow \neg[\kappa_i]\neg\varphi) & \text{(Det)} & \qquad \dfrac{\vdash_{S_i} \varphi}{\vdash_{S_1 + S_2} [\kappa_i]\varphi}\ \text{(N)}. \\ \vdash_{S_1 + S_2} [\kappa_i](\varphi \to \psi) \to ([\kappa_i]\varphi \to [\kappa_i]\psi) & \text{(K)} & \end{array}$$

For an exponent functor:

$$\begin{array}{rll} \vdash_{S^D} [\mathrm{ev}(d)]\varphi \leftrightarrow \neg[\mathrm{ev}(d)]\neg\varphi, & \text{(Det)} & \qquad \dfrac{\vdash_S \varphi}{\vdash_{S^D} [\mathrm{ev}(d)]\varphi}\ \text{(N)}. \\ \vdash_{S^D} [\mathrm{ev}(d)](\varphi \to \psi) \to ([\mathrm{ev}(d)]\varphi \to [\mathrm{ev}(d)]\psi) & \text{(K)} & \end{array}$$

For a powerset functor:

$$\vdash_{\mathcal{P}(S)} [\mathcal{P}](\varphi \to \psi) \to ([\mathcal{P}]\varphi \to [\mathcal{P}]\psi) \quad \text{(K)} \qquad \frac{\vdash_S \varphi}{\vdash_{\mathcal{P}(S)} [\mathcal{P}]\varphi}\ \text{(N)}.$$

We shall write MSML_T for this Many-Sorted Modal Logic associated with the functor T.

Most of the above rules are standard from modal logic, except the rule (DC) — for disjoint cover — used for coproduct ingredients. It says that an element of a coproduct comes from precisely one of the components, see the proof of Lemma 3.5 below.

Lemma 3.3. 1. *Each operator* $O \in \{\mathsf{next}, [\pi_i], [\kappa_i], [ev(d)], [\mathcal{P}]\}$ *maps equivalent formulas to equivalent formulas, and preserves finite conjunctions:*

$$\frac{\vdash_S \varphi \leftrightarrow \psi}{\vdash_S O\varphi \leftrightarrow O\psi} \quad \textit{and} \quad \vdash_S O\top \quad \textit{and} \quad \vdash_S O(\varphi \wedge \psi) \leftrightarrow (O\varphi \wedge O\psi).$$

2. *An operator* $O \in \{\mathsf{next}, [\pi_i], [ev(d)]\}$ *preserves all Boolean operations.*

Proof. 1. The first two statements follow from the rules (N) and (K) for each operator. For preservation of $\wedge$ in the third statement we reason as follows. Since $\varphi \to (\psi \to (\varphi \wedge \psi))$ is a tautology, we have $O(\varphi \to (\psi \to (\varphi \wedge \psi)))$, by what we just proved. By applying axiom (K) twice we get $O\varphi \to (O\psi \to O(\varphi \wedge \psi))$, *i.e.* $(O\varphi \wedge O\psi) \to O(\varphi \wedge \psi)$. For the reverse implication we derive from the tautology $(\varphi \wedge \psi) \to \varphi$ that $O(\varphi \wedge \psi) \to O\varphi$. Similarly, $O(\varphi \wedge \psi) \to O\psi$, and thus $O(\varphi \wedge \psi) \to (O\varphi \wedge O\psi)$.

2. All Boolean structure is preserved because the operators $\mathsf{next}, [\pi_i], [\mathrm{ev}(d)]$ preserve negations, by their axiom (Det).

□

We conclude this section by showing how the formulas of the logic MSML_T can be interpreted in a T-coalgebra.

Definition 3.4. Let $c\colon X \to T(X)$ be a coalgebra for a KPF T. For each ingredient $S \in \mathsf{Ing}(T)$ there is an interpretation function

$$\mathsf{Form}_S \xrightarrow{[\![-]\!]_S} \mathcal{P}(S(X))$$

defined by induction on formulas, using predicate lifting from Definition 2.5:

$$\begin{array}{rcl}
[\![\bot]\!]_S & = & \bot \\
[\![\varphi \to \psi]\!]_S & = & [\![\varphi]\!]_S \to [\![\psi]\!]_S \\
[\![a]\!]_A & = & \{a\} \\
[\![\mathsf{next}\varphi]\!]_{\mathrm{Id}} & = & c^{-1}([\![\varphi]\!]_T) \\
[\![[\pi_i]\varphi]\!]_{S_1 \times S_2} & = & ([\![\varphi]\!]_{S_i})^{\pi_i} \\
[\![[\kappa_i]\varphi]\!]_{S_1 + S_2} & = & ([\![\varphi]\!]_{S_i})^{\kappa_i} \\
[\![[\mathrm{ev}(d)]\varphi]\!]_{S^D} & = & ([\![\varphi]\!]_S)^{\mathrm{ev}(d)} \\
[\![[\mathcal{P}]\varphi]\!]_{\mathcal{P}(S)} & = & ([\![\varphi]\!]_S)^{\mathcal{P}}.
\end{array}$$

Occasionally we shall write $[\![-]\!]^c$ instead of $[\![-]\!]$ to make the coalgebra c explicit. Also, we shall sometimes write $[\![\varphi]\!](x)$ for $x \in [\![\varphi]\!]$, as is usual for predicates.

Notice that state formulas are interpreted as subsets of the state space of a coalgebra.

Once this interpretation has been defined, several "standard" results in the remainder of this section follow easily, see also [23].

Lemma 3.5 (Soundness)**.** *If $\vdash_S \varphi$ in $MSML_T$, then φ is valid, i.e. $[\![\varphi]\!]_S = \top$, in each T-coalgebra.*

Proof. By induction on the length of the derivation of $\vdash_S \varphi$. We only consider the (non-standard) rules (DC) and (Det) for a coproduct functor. The rule (DC) holds because the coprojections κ_i are injective, disjoint and cover the coproduct:

$$\begin{aligned}
&[\![\neg[\kappa_1]\bot \mathrel{\dot\vee} \neg[\kappa_2]\bot]\!] \\
&\quad= \neg(\bot^{\kappa_1}) \mathrel{\dot\vee} \neg(\bot^{\kappa_2}) \\
&\quad= \{z \mid \neg\forall x.\, z = \kappa_1(x) \Rightarrow x \in \bot\} \mathrel{\dot\vee} \{z \mid \neg\forall y.\, z = \kappa_2(y) \Rightarrow y \in \bot\} \\
&\quad= \{z \mid \exists x.\, z = \kappa_1(x)\} \mathrel{\dot\vee} \{z \mid \exists y.\, z = \kappa_2(y)\} \\
&\quad= \top.
\end{aligned}$$

And if $\neg(\bot^{\kappa_i})$ holds of z, *i.e.* if $\exists x.\, z = \kappa_i(x)$, then

$$\begin{aligned}
[\![\neg[\kappa_i]\neg\varphi]\!](z) &\iff \neg\forall x.\, z = \kappa_i(x) \Rightarrow \neg[\![\varphi]\!](x) \\
&\iff \exists x.\, z = \kappa_i(x) \text{ and } [\![\varphi]\!](x) \\
&\iff \forall x.\, z = \kappa_i(x) \Rightarrow [\![\varphi]\!](x) \qquad \text{because } \exists x.\, z = \kappa_i(x) \\
&\iff [\![[\kappa_i]\varphi]\!](z).
\end{aligned}$$

□

Next we intend to show that bisimilar states validate the same formulas.

Lemma 3.6. *Let $c\colon X \to T(X)$ and $d\colon Y \to T(Y)$ be coalgebras for a KPF T, with a bisimulation $R \subseteq X \times Y$. Then for each formula φ of $MSML_T$ of sort $S \in \mathsf{Ing}(T)$, i.e. for each $\varphi \in \mathsf{Form}_S$, we have:*

$$R^S(u, v) \implies \Big([\![\varphi]\!]^c_S(u) \Leftrightarrow [\![\varphi]\!]^d_S(v)\Big).$$

Proof. By induction on the formula φ. For convenience we omit the superscripts "c" and "d".

- $\varphi = \bot$. Obvious, since $[\![\bot]\!] = \bot$.
- $\varphi = \varphi_1 \to \varphi_2$. Assume $R^S(u, v)$ and thus, by (IH), $[\![\varphi_i]\!]_S(u) \Leftrightarrow [\![\varphi_i]\!]_S(v)$. Suppose now that $[\![\varphi_1 \to \varphi_2]\!]_S(u)$, *i.e.* that $[\![\varphi_1]\!]_S(u)$ implies $[\![\varphi_2]\!]_S(u)$. We can then derive $[\![\varphi_1 \to \varphi_2]\!]_S(v)$ as follows.

  $$[\![\varphi_1]\!]_S(v) \Rightarrow [\![\varphi_1]\!]_S(u) \Rightarrow [\![\varphi_2]\!]_S(u) \Rightarrow [\![\varphi_2]\!]_S(v).$$

  The reverse implication is proved similarly.

- $\varphi = a$, and $S = A$. Then $R^A(u,v)$ implies $u =_A v$, and thus:

$$[\![a]\!]_S(u) \Leftrightarrow u = a \Leftrightarrow v = a \Leftrightarrow [\![a]\!]_S(v).$$

- $\varphi = \mathsf{next}\psi$, and $S = \mathrm{Id}$. Assume $R^{\mathrm{Id}}(x,y)$, *i.e.* $R(x,y)$. By induction hypothesis, $R^T(u,v)$ implies $[\![\psi]\!]_T(u) \Leftrightarrow [\![\psi]\!]_T(v)$. Because R is a bisimulation we have $R^T(c(x),c(y))$ and thus

$$[\![\mathsf{next}\psi]\!]_S(x) = [\![\psi]\!]_T(c(x)) \Leftrightarrow [\![\psi]\!]_T(c(y)) = [\![\mathsf{next}\psi]\!]_S(v).$$

- $\varphi = [\pi_i]\psi$, and $S = S_1 \times S_2$. Assume $R^{S_1\times S_2}(u,v)$, so that $R^{S_1}(\pi_1(u),\pi_1(v))$ and $R^{S_2}(\pi_2(u),\pi_2(v))$. Then

$$[\![[\pi_i]\psi]\!]_S(u) = [\![\psi]\!]_{S_i}(\pi_i(u)) \Leftrightarrow [\![\psi]\!]_{S_i}(\pi_i(v)) = [\![[\pi_i]\psi]\!]_S(v).$$

- $\varphi = [\kappa_i]\psi$, and $S = S_1 + S_2$. Assume $R^{S_1\times S_2}(u,v)$, say $u = \kappa_i(x)$, $v = \kappa_i(y)$ with $R^{S_i}(x,y)$. Then

$$[\![[\kappa_i]\psi]\!]_S(u) = [\![\psi]\!]_{S_i}(x) \Leftrightarrow [\![\psi]\!]_{S_i}(y) = [\![[\kappa_i]\psi]\!]_S(v).$$

- $\varphi = [\mathrm{ev}(d)]\psi$, and $S = S_1^D$. Assume $R^{S_1^D}(f,g)$, so that $R^{S_1}(f(d),g(d))$. Then:

$$[\![[\mathrm{ev}(d)]\psi]\!]_S(f) = [\![\psi]\!]_{S_1}(f(d)) \Leftrightarrow [\![\psi]\!]_{S_1}(g(d)) = [\![[\mathrm{ev}(d)]\psi]\!]_S(g).$$

- $\varphi = [\mathcal{P}]\psi$, and $S = \mathcal{P}S_1$. Assume $R^{\mathcal{P}S_1}(\alpha,\beta)$, so that $\forall x \in \alpha.\, \exists y \in \beta.\, R^{S_1}(x,y)$ and $\forall y \in \beta.\, \exists x \in \alpha.\, R^{S_1}(x,y)$. Then

$$[\![[\mathcal{P}]\psi]\!]_S(\alpha) \Leftrightarrow \forall x \in \alpha.\, [\![\psi]\!]_{S_1}(x) \overset{(*)}{\Leftrightarrow} \forall y \in \beta.\, [\![\psi]\!]_{S_1}(y) \Leftrightarrow [\![[\mathcal{P}]\psi]\!]_S(\beta).$$

  The equivalence $\overset{(*)}{\Leftrightarrow}$ requires some care; we show $\overset{(*)}{\Rightarrow}$. Assume $[\![\psi]\!]_{S_1}(x)$ for all $x \in \alpha$, and let $y \in \beta$. Then there is an $x \in \alpha$ with $R^{S_1}(x,y)$. Then $[\![\psi]\!]_{S_1}(y)$ follows from $[\![\psi]\!]_{S_1}(x)$ by (IH).

□

Corollary 3.7. *Bisimilar states in T-coalgebras satisfy the same state formulas of MSML_T:*

$$x \leftrightarroweq y \implies \forall \varphi \in \mathsf{Form}_{Id}.\, [\![\varphi]\!]_{Id}(x) \Leftrightarrow [\![\varphi]\!]_{Id}(y).$$

Later, in Corollary 5.9 we shall see the validity of the reverse implication for *finite* KPFs.

Lemma 3.6 also gives rise to the following (standard) preservation result.

Corollary 3.8. *Consider a KPF T with a homomorphism $\big(X \xrightarrow{c} T(X)\big) \xrightarrow{f} \big(Y \xrightarrow{d} T(Y)\big)$ between two of its coalgebras. Then, for each sort $S \in \mathsf{Ing}(T)$ and formula $\varphi \in \mathsf{Form}_S$,*

$$S(f)^{-1}\big([\![\varphi]\!]^d_S\big) = [\![\varphi]\!]^c_S.$$

As a consequence, all formulas that are valid in d are also valid in c.

Proof. By Lemma 3.6 it suffices to produce a bisimulation $R \subseteq X \times Y$ with $R^S(z, S(f)(z))$, for all $z \in S(X)$. We can take $R = \mathsf{Graph}(f) = \{(x, y) \mid f(x) = y\}$, because $\big(\mathsf{Graph}(f)\big)^S = \mathsf{Graph}(S(f))$. □

4. Many-sorted Boolean algebras with operators

This section introduces the main semantical structures of this paper, namely Boolean algebras with operators which are indexed by sorts. As explained before, these sorts will be ingredients of a given functor. And the indexing by sorts is realised by a functor from sorts to Boolean algebras, like in Proposition 2.7.

Definition 4.1. Let T be a KPF. A *Many-sorted Boolean Algebra with Operators* of type T, or a T-MBAO for short, consists of a "sort-indexed Boolean algebra"

$$\mathbf{Ing}(T)^{\mathrm{op}} \xrightarrow{\quad\Phi\quad} \mathbf{BA}_{\wedge}$$

such that

1. the functions $\Phi(\pi_i)$ and $\Phi(\mathrm{ev}(d))$ induced by projection and evaluation paths preserve all Boolean operations;
2. the functions $\Phi(\kappa_i)$ induced by coprojection paths satisfy

$$\begin{array}{rcl} \neg\Phi(\kappa_1)(\bot) \mathbin{\dot\vee} \neg\Phi(\kappa_2)(\bot) & = & \top \\ \neg\Phi(\kappa_i)(\bot) & \leq & \neg\Phi(\kappa_i)(\neg\alpha) \leftrightarrow \Phi(\kappa_i)(\alpha). \end{array}$$

Together with the following additional structure.

3. For each constant functor $A \in \mathsf{Ing}(T)$ a map $\mathsf{obs}_A \colon A \to \Phi(A)$ satisfying $\dot{\bigvee}_{a \in A} \mathsf{obs}_A(a) = \top$. Note that finiteness of A is needed for this disjunction to exist.
4. If the identity functor Id is in $\mathsf{Ing}(T)$, a mapping $\mathsf{next} \colon \Phi(T) \to \Phi(\mathrm{Id})$ which preserves all Boolean operations.

Before we consider examples of MBAOs, we show how to interpret the many-sorted modal logic from the previous section in an arbitrary MBAO.

Definition 4.2. Let $\Phi = (\Phi \colon \mathbf{Ing}(T)^{\mathrm{op}} \to \mathbf{BA}_{\wedge}, \mathsf{obs}, \mathsf{next})$ be a T-MBAO as above. An interpretation $[\![-]\!]$ in Φ of the many-sorted modal logic MSML_T associated with the functor T is introduced *via* interpretation functions

$$\mathsf{Form}_S \xrightarrow{\quad[\![-]\!]_S\quad} \Phi(S)$$

which are defined inductively:

$$\begin{aligned}
[\![\bot]\!]_S &= \bot \\
[\![\varphi \to \psi]\!]_S &= [\![\varphi]\!]_S \to [\![\psi]\!]_S \\
[\![a]\!]_A &= \mathsf{obs}_A(a) \\
[\![\mathsf{next}\varphi]\!]_{\mathrm{Id}} &= \mathsf{next}([\![\varphi]\!]_T) \\
[\![[\pi_i]\varphi]\!]_{S_1 \times S_2} &= \Phi(\pi_i)([\![\varphi]\!]_{S_i}) \\
[\![[\kappa_i]\varphi]\!]_{S_1 + S_2} &= \Phi(\kappa_i)([\![\varphi]\!]_{S_i}) \\
[\![[\mathrm{ev}(a)]\varphi]\!]_{S^D} &= \Phi(\mathrm{ev}(d))([\![\varphi]\!]_S) \\
[\![[\mathcal{P}]\varphi]\!]_{\mathcal{P}S} &= \Phi(\mathcal{P})([\![\varphi]\!]_S).
\end{aligned}$$

Sometimes we shall write $[\![-]\!]^\Phi$ instead of $[\![-]\!]$ to make the MBAO Φ explicit.

Lemma 4.3 (Soundness)**.** *If $\vdash_S \varphi$ in MSML$_T$, then φ is valid, i.e. $[\![\varphi]\!]_S = \top$, in each T-MBAO.*

Proof. By induction on the length of the derivation of $\vdash_S \varphi$, using properties 1–4 from Definition 4.1. □

From the formulas of our logic we can also construct a model, in a so-called Lindenbaum construction. This syntactic model has some special properties, see Propositions 4.5 and 4.8. It will be used later in Section 5.3 to construct final coalgebras.

Example 4.4. In the logic MSML$_T$ for a KPF T, we define an equivalence relation $\sim_S$ on Form_S, for an ingredient $S \in \mathsf{Ing}(T)$, by:

$$\varphi \sim_S \psi \iff^{\mathrm{def}} \vdash_S \varphi \leftrightarrow \psi.$$

The resulting quotient $\mathsf{Form}_S/{\sim_S} = \{ |\, \varphi \,|_{\sim_S} \mid \varphi \in \mathsf{Form}_S \}$ then forms a Boolean algebra, with obvious structure defined *via* representatives:

$$\top = |\, \top \,|_{\sim_S}, \quad \neg |\, \varphi \,|_{\sim_S} = |\, \neg\varphi \,|_{\sim_S}, \quad |\, \varphi \,|_{\sim_S} \wedge |\, \psi \,|_{\sim_S} = |\, \varphi \wedge \psi \,|_{\sim_S}, \text{ etc.}$$

In this way we get the object part $S \mapsto \mathsf{Form}_S/{\sim_S}$ of a functor $\mathbf{Ing}(T)^{\mathrm{op}} \to \mathbf{BA}_\wedge$, for which we shall write $\mathcal{L}_T$. For the morphism part of $\mathcal{L}_T$, consider a path $p = s_1 \cdots s_n$ where each individual step s_i is of the form $\pi_i, \kappa_i, \mathrm{ev}(d)$ or $\mathcal{P}$, say in $S_1 \overset{p}{\rightsquigarrow} S_2$. We then define $\mathcal{L}_T(p) \colon \mathsf{Form}_{S_2}/{\sim_{S_2}} \to \mathsf{Form}_{S_1}/{\sim_{S_1}}$ by $|\, \varphi \,|_{\sim_{S_2}} \mapsto |\, [s_1] \cdots [s_n]\varphi \,|_{\sim_{S_1}}$. In this way we get a functor which satisfies requirements (1) and (2) from Definition 4.1, by Lemma 3.3.

The definition of an MBAO requires two special functions. The first one, namely $\mathsf{obs}_A \colon A \to \mathcal{L}_T(A) = \mathsf{Form}_A/{\sim_A}$, is simply $a \mapsto |\, a \,|_{\sim_A}$. The other function, $\mathsf{next} \colon \mathcal{L}_T(T) \to \mathcal{L}_T(\mathrm{Id})$, is the function $\mathsf{Form}_T/{\sim_T} \to \mathsf{Form}_{\mathrm{Id}}/{\sim_{\mathrm{Id}}}$ given by $|\, \varphi \,|_{\sim_T} \mapsto |\, \mathsf{next}\varphi \,|_{\sim_{\mathrm{Id}}}$.

Proposition 4.5 (Completeness for MBAOs)**.** *If a formula in $MSML_T$ is valid in all T-MBAOs, then it is derivable.*

Proof. If $\varphi \in \mathsf{Form}_S$ is valid in each T-MBAO, it is in particular valid in the Lindenbaum model $\mathcal{L}_T \colon \mathbf{Ing}(T)^{\mathrm{op}} \to \mathbf{BA}_\wedge$ from the previous example. The interpretation in this model is given by: $[\![\varphi]\!]_S = |\, \varphi \,|_{\sim_S}$. Validity of $\varphi \in \mathsf{Form}_S$ means that $|\, \varphi \,|_{\sim_S} = |\, \top \,|_{\sim_S}$, *i.e.* that $\vdash_S \varphi \leftrightarrow \top$, and thus that $\vdash_S \varphi$. □

We continue with an important construction of MBAOs, namely from coalgebras. The construction is already suggested by Proposition 2.7 and by the interpretation of the logic in a coalgebra, at the end of the previous section.

Example 4.6. Assume a coalgebra $c \colon X \to T(X)$ for a KPF T. As we saw in Proposition 2.7, predicate lifting gives rise to a functor:

$$\mathbf{Ing}(T)^{\mathrm{op}} \xrightarrow{\quad \mathcal{A}(c) \quad} \mathbf{BA}_\wedge$$

by $S \mapsto \mathcal{P}(S(X))$ and $p \mapsto (-)^p$. This functor is actually a T-MBAO, because the four requirements of Definition 4.1 are satisfied:

1. the predicate lifting functions $\mathcal{A}(c)(\pi_i) = (-)^{\pi_i}$ and $\mathcal{A}(c)(\mathrm{ev}(d)) = (-)^{\mathrm{ev}(d)}$ preserve all Boolean structure, see Lemma 2.6(1);
2. the functions $\mathcal{A}(c)(\kappa_i) = (-)^{\kappa_i}$ satisfy the required properties, as shown in the proof of the soundness Lemma 3.5;
3. for each constant functor $A \in \mathsf{Ing}(T)$, there is a canonical function $\mathsf{obs}_A = \{-\} \colon A \to \mathcal{A}(c)(A) = \mathcal{P}(A)$. It clearly satisfies $\dot{\bigvee}_{a \in A} \mathsf{obs}_A(a) = \top$;
4. if $\mathrm{Id} \in \mathsf{Ing}(T)$ then there is function next from $\mathcal{A}(c)(T)$ to $\mathcal{A}(c)(\mathrm{Id})$, *i.e.* from $\mathcal{P}(T(X))$ to $\mathcal{P}(X)$, namely

$$\mathsf{next}(\alpha) = c^{-1}(\alpha) = \{x \mid c(x) \in \alpha\}.$$

   It commutes with all the Boolean operations — like any inverse image function.

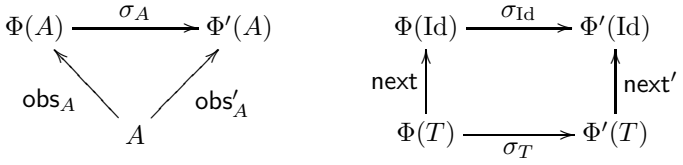
Notice that this MBAO $\mathcal{A}(c)$ constructed out of the coalgebra c makes use of the operations used in the interpretation of many-sorted modal logic in a coalgebra, see Definition 3.4. Indeed, it is not hard to see that the interpretation $[\![\varphi]\!]^c$ of a formula φ in the coalgebra c is the same as its interpretation $[\![\varphi]\!]^{\mathcal{A}(c)}$ in the associated MBAO $\mathcal{A}(c)$.

The next step is to consider morphisms of MBAOs.

Definition 4.7. A homomorphism from one T-MBAOs $(\Phi, \mathsf{obs}, \mathsf{next})$ to another $(\Phi', \mathsf{obs}', \mathsf{next}')$ is a natural transformation σ in:

$$\mathbf{Ing}(T)^{\mathrm{op}} \underset{\Phi'}{\overset{\Phi}{\Downarrow \sigma}} \mathbf{BA}_\wedge$$

such that for each ingredient $S \in \mathsf{Ing}(T)$ the component $\sigma_S \colon \Phi(S) \to \Phi'(S)$ preserves (all) the Boolean structure, and such that for all appropriate ingredients, the following two diagrams commute.



This yields a category, for which we shall write $\mathbf{MBAO}(T)$.

Now that we have maps between MBAOs we can establish a familiar property of Lindenbaum constructions.

Proposition 4.8. *The Lindenbaum MBAO $\mathcal{L}_T$ from Example 4.4 is an initial object in the category $\boldsymbol{MBAO}(T)$: for an arbitrary T-MBAO Φ there is a unique homomorphism $[\![-]\!] \colon \mathcal{L}_T \to \Phi$. It corresponds to the interpretation of $MSML_T$ in Φ, as described in Definition 4.2.*

This view of interpretations as structure preserving maps comes from Lawvere's so-called functorial semantics, see [18].

Proof. First we note that the interpretation functions $[\![-]\!]_S \colon \mathsf{Form}_S \to \Phi(S)$ from Definition 4.2 map the equivalence relation $\sim_S$ from Example 4.4 to equality, since if $\vdash_S \varphi \leftrightarrow \psi$, then $([\![\varphi]\!]_S \leftrightarrow [\![\psi]\!]_S) = \top$ in the Boolean algebra $\Phi(S)$, by soundness, and thus $[\![\varphi]\!]_S = [\![\psi]\!]_S$. This means that these interpretation functions $\mathsf{Form}_S \to \Phi(S)$ give rise to functions $\mathcal{L}_T(S) = \mathsf{Form}_S/{\sim_S} \to \Phi(S)$ preserving the Boolean algebra structure. They form the components of a natural transformation $\mathcal{L}_T \Rightarrow \Phi$, which is a homomorphism of T-MBAOs. It is not hard to see that it is the only possible one, because it is completely determined by the requirements in Definition 4.7. □

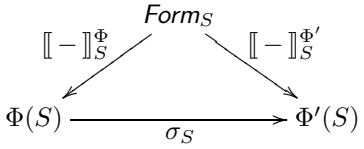
Proposition 4.9. *The construction $c \mapsto \mathcal{A}(c)$ from Example 4.6 yields a functor*

$$\boldsymbol{CoAlg}(T)^{op} \xrightarrow{\quad\mathcal{A}\quad} \boldsymbol{MBAO}(T).$$

Proof. For a morphism of coalgebras $(X \xrightarrow{c} T(X)) \xrightarrow{f} (Y \xrightarrow{d} T(Y))$ we obtain a natural transformation $\mathcal{A}(f) \colon \mathcal{A}(d) \Rightarrow \mathcal{A}(c)$ with components $\mathcal{A}(f)_S$ at an ingredient $S \in \mathsf{Ing}(T)$ defined as $S(f)^{-1}$. Naturality follows from Lemma 2.6(3). The diagrams in Definition 4.7 obviously commute. □

Finally, the following point can be mentioned.

Lemma 4.10. *A homomorphism $\sigma\colon \Phi \Rightarrow \Phi'$ of T-MBAOs preserves interpretations, in the sense that the diagram*



commutes for each sort $S \in \mathsf{Ing}(T)$.

5. From MBAOs to coalgebras

In the previous section we have seen a translation from coalgebras to MBAOs. The aim in this section is to study reverse translations, and their consequences. One such translation arises as an algebraic reformulation of the construction used by Rößiger in [23] (Def. 5.8), formulated in terms of maximally consistent sets of formulas, and used for a completeness result for dynamic models (coalgebras) of many-sorted modal logic. Here we shall identify an essential step from this construction (in the next definition) and use it for an alternative translation. The latter can also be used for a completeness result. It turns out to be more natural because it gives rise to an ultrafilter extension result (Sect. 5.2) and a final coalgebra (Sect. 5.3).

Definition 5.1. Let Φ be an MBAO for a KPF T with the identity functor Id as ingredient. For each sort $S \in \mathsf{Ing}(T)$ there is a canonical map

$$\mathsf{spec}\,\Phi(S) \xrightarrow{\mathsf{r}_\Phi(S)} S\big(\mathsf{spec}\,\Phi(\mathrm{Id})\big)$$

which produces S-structure from ultrafilters. It is defined in the following way.

$$\begin{array}{rcl}
\mathsf{r}_\Phi(A)(U) & = & a \ \text{ if and only if } \ \mathsf{obs}(a) \in U \\
\mathsf{r}_\Phi(\mathrm{Id})(U) & = & U \\
\mathsf{r}_\Phi(S_1 \times S_2)(U) & = & \langle \mathsf{r}_\Phi(S_1)\big(\Phi(\pi_1)^{-1}(U)\big),\, \mathsf{r}_\Phi(S_2)\big(\Phi(\pi_2)^{-1}(U)\big)\rangle \\
\mathsf{r}_\Phi(S_1 + S_2)(U) & = & \begin{cases} \kappa_1 \mathsf{r}_\Phi(S_1)\big(\Phi(\kappa_1)^{-1}(U)\big) & \text{if } \neg\Phi(\kappa_1)(\bot) \in U \\ \kappa_2 \mathsf{r}_\Phi(S_2)\big(\Phi(\kappa_2)^{-1}(U)\big) & \text{if } \neg\Phi(\kappa_2)(\bot) \in U \end{cases} \\
\mathsf{r}_\Phi(S^D)(U) & = & \lambda d \in D.\, \mathsf{r}_\Phi(S)\big(\Phi(\mathrm{ev}(d))^{-1}(U)\big) \\
\mathsf{r}_\Phi(\mathcal{P}S)(U) & = & \{\mathsf{r}_\Phi(S)(V) \mid V \in \mathsf{spec}\,\Phi(S) \text{ and } \Phi(\mathcal{P})^{-1}(U) \subseteq V\}.
\end{array}$$

These maps $\mathsf{r}_\Phi(S)$ do the crucial work of extracting the structure of the functor S from ultrafilters $U \in \Phi(S)$. It is not hard to see that they are well-defined. In the constant functor case we use requirement (3) from Definition 4.1, which says that $\dot{\bigvee}_{a\in A}\mathsf{obs}_A(a) = \top$, and thus an element of an ultrafilter $U \in \Phi(A)$. As a

result, there is precisely one $a \in A$ with $\mathsf{obs}_A(a) \in U$. In the identity, product and exponent case we use that the functions next, $\Phi(\pi_i)$ and $\Phi(\mathrm{ev}(a))$ preserve all Boolean operations, so that their inverse image functors map ultrafilters to ultrafilters. In the coproduct case we use the disjoint cover property $\neg\Phi(\kappa_1)(\bot) \dot\vee \neg\Phi(\kappa_2)(\bot) = \top \in U$ to make a case distinction. In the case $\neg\Phi(\kappa_i)(\bot) \in U$, we then use $\neg\Phi(\kappa_i)(\neg\alpha) \in U$ if and only if $\Phi(\kappa_i)(\alpha) \in U$, in order to see that $\Phi(\kappa_i)^{-1}(U)$ is an ultrafilter.

The diligent reader may have noticed that the next map from the MBAO is not used in the above construction. Indeed, its role is separated from these maps $\mathsf{r}_\Phi(S)$, but is crucially used to construct coalgebras as follows.

Definition 5.2. Consider a T-MBAO Φ as above. It gives rise to two T-coalgebras, one with state space $\mathsf{spec}\,\Phi(T)$ and one with $\mathsf{spec}\,\Phi(\mathrm{Id})$.

1. Rößiger's construction ([23], Def. 5.8) yields a coalgebra:

$$\mathcal{R}(\Phi) = \Big(\mathsf{spec}\,\Phi(T) \xrightarrow{\mathsf{r}_\Phi(T)} T\big(\mathsf{spec}\,\Phi(\mathrm{Id})\big) \xrightarrow{T(\mathsf{next}^{-1})} T\big(\mathsf{spec}\,\Phi(T)\big)\Big).$$

2. Alternatively, one can define:

$$\mathcal{C}(\Phi) = \Big(\mathsf{spec}\,\Phi(\mathrm{Id}) \xrightarrow{\mathsf{next}^{-1}} \mathsf{spec}\,\Phi(T) \xrightarrow{\mathsf{r}_\Phi(T)} T\big(\mathsf{spec}\,\Phi(\mathrm{Id})\big)\Big).$$

By construction, next^{-1} is a homomorphism of coalgebras $\mathcal{C}(\Phi) \to \mathcal{R}(\Phi)$.

Proposition 5.3. *Both the mappings $\Phi \mapsto \mathcal{R}(\Phi)$ and $\Phi \mapsto \mathcal{C}(\Phi)$ are functorial, and next^{-1} is a natural transformation between them:*

$$\boldsymbol{MBAO}(T) \underset{\mathcal{R}}{\overset{\mathcal{C}}{\Downarrow \mathsf{next}^{-1}}} \boldsymbol{CoAlg}(T)^{op}.$$

Proof. Let $\sigma\colon \Phi \to \Psi$ be a homomorphism of T-MBAOs. One can define $\mathcal{R}$ on a morphism σ as σ_T^{-1}, and $\mathcal{C}$ as $\sigma_{\mathrm{Id}}^{-1}$. This yields homorphisms of coalgebras, because for each ingredient $S \in \mathsf{Ing}(T)$ the following diagram commutes.

$$\begin{array}{ccc} S\big(\mathsf{spec}\,\Psi(\mathrm{Id})\big) & \xrightarrow{S(\sigma_{\mathrm{Id}}^{-1})} & S\big(\mathsf{spec}\,\Phi(\mathrm{Id})\big) \\ \uparrow{\scriptstyle \mathsf{r}_\Psi(S)} & & \uparrow{\scriptstyle \mathsf{r}_\Phi(S)} \\ \mathsf{spec}\,\Psi(S) & \xrightarrow[\sigma_S^{-1}]{} & \mathsf{spec}\,\Phi(S) \end{array}$$

The proof proceeds by induction on the structure of S, and is straightforward except for the powerset case. So assume $S = \mathcal{P}S_1$ and $U \in \mathsf{spec}\,\Psi(S)$. Then:

$$\begin{array}{rcl}
\big(S(\sigma_{\mathrm{Id}}^{-1}) \circ \mathsf{r}_\Psi(S)\big)(U) & = & \{S_1(\sigma_{\mathrm{Id}}^{-1})(\mathsf{r}_\Psi(S_1)(V)) \mid \Psi(\mathcal{P})^{-1}(U) \subseteq V\} \\
& \overset{\text{(IH)}}{=} & \{\mathsf{r}_\Phi(S_1)(\sigma_{S_1}^{-1}(V)) \mid \Psi(\mathcal{P})^{-1}(U) \subseteq V\} \\
& \overset{(*)}{=} & \{\mathsf{r}_\Phi(S_1)(W) \mid \Phi(\mathcal{P})^{-1}(\sigma_S^{-1}(U)) \subseteq W\} \\
& = & \big(\mathsf{r}_\Phi(S) \circ \sigma_S^{-1}\big)(U).
\end{array}$$

The inclusion-part ($\subseteq$) of the marked equation $\overset{(*)}{=}$ is obvious, so we concentrate on ($\supseteq$). Let therefore $W \in \mathsf{spec}\,\Phi(S_1)$ be given with $\Phi(\mathcal{P})^{-1}(\sigma_S^{-1}(U)) \subseteq W$. In order to get an appropriate V, we follow the lines of the proof of Lemma 2.11, and consider the filter

$$F = \uparrow\!\big(\Psi(\mathcal{P})^{-1}(U) \cup \sigma_{S_1}(W)\big).$$

The first step is to show that $\bot \notin F$. If not, *i.e.* if $\bot \in F$, then there are $x \in \Psi(\mathcal{P})^{-1}(U)$ and $y \in W$ with $\bot = x \wedge \sigma_{S_1}(y)$. Then $x \leq \sigma_{S_1}(\neg y)$, which yields $\Psi(\mathcal{P})(x) \leq \Psi(\mathcal{P})(\sigma_{S_1}(\neg y)) = \sigma_S(\Phi(\mathcal{P})(\neg y))$, and thus $\sigma_S(\Phi(\mathcal{P})(\neg y)) \in U$. But then $\neg y \in \Phi(\mathcal{P})^{-1}(\sigma_S^{-1}(U)) \subseteq W$, which contradicts $y \in W$.

Lemma 2.10 now yields an ultrafilter $V \in \mathsf{spec}\,\Phi(S_1)$ with $\Psi(\mathcal{P})^{-1}(U) \subseteq V$ and $\sigma_{S_1}(W) \subseteq V$. The latter yields $W = \sigma_{S_1}^{-1}(V)$, and thus shows that V is the ultrafilter we are looking for. The inclusion-part ($\subseteq$) of $W = \sigma_{S_1}^{-1}(V)$ is obvious. For ($\supseteq$), notice that if $y \in \sigma_{S_1}^{-1}(V)$ but $y \notin W$, then $\neg y \in W$, and thus $\neg y \in \sigma_{S_1}^{-1}(V)$. This is impossible because $\sigma_{S_1}^{-1}(V)$ is an ultrafilter. □

From a logical perspective, the $\mathcal{C}$ functor takes states of the coalgebra that is constructed to be ultrafilters of *state* formulas. This is more natural than the $\mathcal{R}$ functor, and clearly in line with the standard approach in modal logic, see Section 2.4. Also, it gives rise to the results below.

5.1. Completeness

Towards the end of Example 4.6 we saw that the interpretation of a formula in a coalgebra c is the same as its interpretation in its associated MBAO $\mathcal{A}(c)$. The relation between interpretations in an MBAO Φ and the translations $\mathcal{R}(\Phi)$ and $\mathcal{C}(\Phi)$ are more complicated, but follows a standard Łoś-style pattern.

Lemma 5.4. *Let Φ be a T-MBAO, and φ a formula of sort $S \in \mathsf{Ing}(T)$. Then for an ultrafilter $U \in \mathsf{spec}\,\Phi(S)$,*

$$[\![\varphi]\!]^\Phi_S \in U \overset{(1)}{\iff} r_\Phi(S)(U) \in [\![\varphi]\!]^{\mathcal{C}(\Phi)}_S \overset{(2)}{\iff} S(\mathsf{next}^{-1})(r_\Phi(S)(U)) \in [\![\varphi]\!]^{\mathcal{R}(\Phi)}_S.$$

Proof. The second equivalence $\overset{(2)}{\iff}$ follows directly from Corollary 3.8, so we concentrate on $\overset{(1)}{\iff}$. It is proved by induction on the structure of φ.

- The case $\varphi = \bot$ is obvious, for each sort.

- Similarly, if $\varphi = \varphi_1 \to \varphi_2$, then

$$\begin{aligned}
&[\![\varphi_1 \to \varphi_2]\!]^\Phi_S = [\![\varphi_1]\!]^\Phi_S \to [\![\varphi_2]\!]^\Phi_S \in U \\
&\quad\iff\quad [\![\varphi_1]\!]^\Phi_S \in U \text{ implies } [\![\varphi_2]\!]^\Phi_S \in U \\
&\quad\overset{\text{(IH)}}{\iff}\quad \mathsf{r}_\Phi(S)(U) \in [\![\varphi_1]\!]^{\mathcal{C}(\Phi)}_S \text{ implies } \mathsf{r}_\Phi(S)(U) \in [\![\varphi_2]\!]^{\mathcal{C}(\Phi)}_S \\
&\quad\iff\quad \mathsf{r}_\Phi(S)(U) \in [\![\varphi_1 \to \varphi_2]\!]^{\mathcal{C}(\Phi)}_S.
\end{aligned}$$

- If $\varphi = a \in A$, when S is a constant functor A, we get

$$[\![\varphi]\!]^\Phi_S = \mathsf{obs}_S(a) \in U \quad\iff\quad \mathsf{r}_\Phi(S)(U) = a \in \{a\} = [\![\varphi]\!]^{\mathcal{C}(\Phi)}_S.$$

- If $\varphi = \mathsf{next}\psi$ when S is the identity functor Id, then:

$$\begin{aligned}
&[\![\mathsf{next}\psi]\!]^\Phi_S = \mathsf{next}([\![\psi]\!]^\Phi_T) \in U \\
&\quad\iff\quad [\![\psi]\!]^\Phi_T \in \mathsf{next}^{-1}(U) \\
&\quad\overset{\text{(IH)}}{\iff}\quad \mathsf{r}_\Phi(T)(\mathsf{next}^{-1}(U)) = \mathcal{C}(\Phi)(U) \in [\![\psi]\!]^{\mathcal{C}(\Phi)}_T \\
&\quad\iff\quad \mathsf{r}_\Phi(S)(U) = U \in \mathcal{C}(\Phi)^{-1}([\![\psi]\!]^{\mathcal{C}(\Phi)}_T) = [\![\mathsf{next}\psi]\!]^{\mathcal{C}(\Phi)}_S.
\end{aligned}$$

- If $\varphi = [\pi_i]\psi$ when S is a product functor $S_1 \times S_2$, we have:

$$\begin{aligned}
&[\![[\pi_i]\psi]\!]^\Phi_S = \Phi(\pi_i)([\![\psi]\!]^\Phi_{S_i}) \in U \\
&\quad\iff\quad [\![\psi]\!]^\Phi_{S_i} \in \Phi(\pi_i)^{-1}(U) \\
&\quad\overset{\text{(IH)}}{\iff}\quad \mathsf{r}_\Phi(S_i)(\Phi(\pi_i)^{-1}(U)) \in [\![\psi]\!]^{\mathcal{C}(\Phi)}_{S_i} \\
&\quad\iff\quad \mathsf{r}_\Phi(S)(U) = \langle \mathsf{r}_\Phi(S_1)(\Phi(\pi_1)^{-1}(U)),\, \mathsf{r}_\Phi(S_2)(\Phi(\pi_2)^{-1}(U)) \rangle \\
&\qquad\qquad \in ([\![\psi]\!]^{\mathcal{C}(\Phi)}_{S_i})^{\pi_i} = [\![[\pi_i]\psi]\!]^{\mathcal{C}(\Phi)}_S.
\end{aligned}$$

- If $\varphi = [\kappa_i]\psi$ when S is a coproduct functor $S_1 + S_2$:

$$\begin{aligned}
&[\![[\kappa_i]\psi]\!]^\Phi_S = \Phi(\kappa_i)([\![\psi]\!]^\Phi_{S_i}) \in U \\
&\quad\overset{(*)}{\iff}\quad \neg\Phi(\kappa_i)(\bot) \in U \text{ implies } [\![\psi]\!]^\Phi_{S_i} \in \Phi(\kappa_i)^{-1}(U) \\
&\quad\overset{\text{(IH)}}{\iff}\quad \neg\Phi(\kappa_i)(\bot) \in U \text{ implies } \mathsf{r}_\Phi(S_i)(\Phi(\kappa_i)^{-1}(U)) \in [\![\psi]\!]^{\mathcal{C}(\Phi)}_{S_i} \\
&\quad\iff\quad \mathsf{r}_\Phi(S)(U) \in ([\![\psi]\!]^{\mathcal{C}(\Phi)}_{S_i})^{\kappa_i} = [\![[\kappa_i]\psi]\!]^{\mathcal{C}(\Phi)}_S.
\end{aligned}$$

  The marked implication $\overset{(*)}{\Longrightarrow}$ obviously holds. For $\overset{(*)}{\Longleftarrow}$, we distinguish whether $\neg\Phi(\kappa_i)(\bot) \in U$ or not. In the first case we are done, and in the second case we know $\Phi(\kappa_i)(\bot) \in U$ and thus $\Phi(\kappa_i)([\![\psi]\!]^\Phi_{S_i}) \in U$ because $\Phi(\kappa_i)(\bot) \leq \Phi(\kappa_i)([\![\psi]\!]^\Phi_{S_i})$ by montonicity of $\Phi(\kappa_i)$.
- The case when $\varphi = [\mathrm{ev}(d)]\psi$ is much like the projection case, and will therefore be skipped.

- Finally, if $\varphi = [\mathcal{P}]\psi$ we use Lemma 2.11:

$$\begin{aligned}
&[\![[\mathcal{P}]\psi]\!]^{\Phi}_{S} = \Phi(\mathcal{P})([\![\psi]\!]^{\Phi}_{\mathcal{P}S}) \in U \\
&\quad\iff [\![\psi]\!]^{\Phi}_{\mathcal{P}S} \in \Phi(\mathcal{P})^{-1}(U) = \bigcap\{V \in \mathsf{spec}\,\Phi(\mathcal{P}S) \mid \Phi(\mathcal{P})^{-1}(U) \subseteq V\} \\
&\quad\iff \forall V \in \mathsf{spec}\,\Phi(\mathcal{P}S).\, \Phi(\mathcal{P})^{-1}(U) \subseteq V \text{ implies } [\![\psi]\!]^{\Phi}_{\mathcal{P}S} \in V \\
&\quad\overset{\text{(IH)}}{\iff} \forall V \in \mathsf{spec}\,\Phi(\mathcal{P}S).\, \Phi(\mathcal{P})^{-1}(U) \subseteq V \text{ implies } \mathsf{r}_{\Phi}(\mathcal{P}S)(V) \in [\![\psi]\!]^{\mathcal{C}(\Phi)}_{\mathcal{P}S} \\
&\quad\iff \mathsf{r}_{\Phi}(S)(U) = \{\mathsf{r}_{\Phi}(\mathcal{P}S)(V) \mid \Phi(\mathcal{P})^{-1}(U) \subseteq V\} \subseteq [\![\psi]\!]^{\mathcal{C}(\Phi)}_{\mathcal{P}S} \\
&\quad\iff \mathsf{r}_{\Phi}(S)(U) \in ([\![\psi]\!]^{\mathcal{C}(\Phi)}_{\mathcal{P}S})^{\mathcal{P}} = [\![[\mathcal{P}]\psi]\!]^{\mathcal{C}(\Phi)}_{S}.
\end{aligned}$$

□

This technical lemma is crucial for the following result from [23].

Theorem 5.5 (Completeness for coalgebras)**.** *If a formula in $MSML_T$ is valid in all T-coalgebras, then it is derivable.*

The proof in [23] makes use of the coalgebra $\mathcal{R}(\mathcal{L}_T)$ obtained by applying the functor $\mathcal{R}$ to the Lindenbaum MBAO $\mathcal{L}_T$ from Example 4.4. As we show, also $\mathcal{C}(\mathcal{L}_T)$ can be used.

Proof. Assume that a formula φ of sort $S \in \mathsf{Ing}(T)$ holds in each coalgebra. In particular this means that both $[\![\varphi]\!]^{\mathcal{R}(\mathcal{L}_T)}_{S} = \top$ and $[\![\varphi]\!]^{\mathcal{C}(\mathcal{L}_T)}_{S} = \top$. From either of those facts one can conclude that $|\varphi|_{\sim_S} = [\![\varphi]\!]^{\mathcal{L}_T}_{S} \in U$, for every ultrafilter $U \in \mathsf{spec}\,\mathcal{L}_T(S)$, by Lemma 5.4. Hence $|\varphi|_{\sim_S} = |\top|_{\sim_S}$, by Corollary 2.12, which says that $\vdash_S \varphi \leftrightarrow \top$, and thus that $\vdash_S \varphi$. □

5.2. Ultrafilter extensions for coalgebras

In this subsection we establish an ultrafilter extension result for coalgebras of *finite* KPFs. Recall that for an arbitrary set X, there is the so-called ultrafilter extension map $\varepsilon_X \colon X \to \mathsf{spec}\,\mathcal{P}X$ sending $x \in X$ to the principal ultrafilter $\uparrow\{x\} = \{\alpha \in \mathcal{P}X \mid x \in \alpha\}$. Our aim is to show that if X carries a coalgebra structure, then this ε is a homomorphism of coalgebras.

Lemma 5.6. *Let $c \colon X \to T(X)$ be a coalgebra for a finite KPF T, with associated MBAO $\mathcal{A}(c)$ and canonical map $\mathsf{r}_{\mathcal{A}(c)}(S)$ as in Definition 5.1. For each sort $S \in \mathsf{Ing}(T)$ the following diagram commutes.*

$$\begin{array}{ccc}
\mathsf{spec}\,\mathcal{P}(S(X)) = \mathsf{spec}\,\mathcal{A}(c)(S) & \xrightarrow{\mathsf{r}_{\mathcal{A}(c)}(S)} & S\big(\mathsf{spec}\,\mathcal{A}(c)(Id)\big) = S\big(\mathsf{spec}\,\mathcal{P}(X)\big). \\
\uparrow{\scriptstyle \varepsilon_{S(X)}} & & \uparrow{\scriptstyle S(\varepsilon_X)} \\
S(X) & & S(X)
\end{array}$$

Proof. By induction on the structure of S.

- If S is a constant functor A, then

$$\mathsf{r}_{\mathcal{A}(c)}(A)(\varepsilon_A(a)) = b \iff \{b\} \in \varepsilon_A(a) \iff a \in \{b\} \iff a = b.$$

- If S is the identity functor, the result obviously holds.
- If $S = S_1 \times S_2$, then we first note that

$$\begin{array}{rcl} \mathcal{A}(c)(\pi_i)^{-1}(\varepsilon_{S(X)}(z)) & = & \{\alpha \in \mathcal{P}(S_i(X)) \mid \alpha^{\pi_i} \in \varepsilon_{S_i(X)}(z)\} \\ & = & \{\alpha \in \mathcal{P}(S_i(X)) \mid z \in \alpha^{\pi_i}\} \\ & = & \{\alpha \in \mathcal{P}(S_i(X)) \mid \pi_i(z) \in \alpha\} \\ & = & \varepsilon_{S_i(X)}(\pi_i(z)). \end{array}$$

It allows us to prove:

$$\begin{array}{rcl} & & \mathsf{r}_{\mathcal{A}(c)}(S)(\varepsilon_{S(X)}(z)) \\ & = & \langle \mathsf{r}_{\mathcal{A}(c)}(S_1)(\mathcal{A}(c)(\pi_1)^{-1}(\varepsilon_{S(X)}(z))),\ \mathsf{r}_{\mathcal{A}(c)}(S_2)(\mathcal{A}(c)(\pi_2)^{-1}(\varepsilon_{S(X)}(z)))\rangle \\ & = & \langle \mathsf{r}_{\mathcal{A}(c)}(S_1)(\varepsilon_{S_1(X)}(\pi_1(z))),\ \mathsf{r}_{\mathcal{A}(c)}(S_2)(\varepsilon_{S_2(X)}(\pi_2(z)))\rangle \\ & \overset{\text{(IH)}}{=} & \langle S_1(\varepsilon_X(\pi_1(z))),\ S_2(\varepsilon_X(\pi_2(z)))\rangle \\ & = & S(\varepsilon_X)(z). \end{array}$$

- If $S = S_1 + S_2$ then we use that:

$$\begin{array}{rcl} \neg\mathcal{A}(c)(\kappa_i)(\bot) \in \varepsilon_{S(X)}(z) & \iff & z \notin \bot^{\kappa_i} \\ & \iff & \neg\forall u \in S_i(X).\, \kappa_i(z) = u \Rightarrow y \in \bot \\ & \iff & \exists u \in S_i(X).\, \kappa_i(z) = u. \end{array}$$

And in that case: $\mathcal{A}(c)(\kappa_i)^{-1}(\varepsilon_{S(X)}(\kappa_i(u)) = \varepsilon_{S_i(X)}(u)$. Thus:

$$\begin{array}{rcll} & & \mathsf{r}_{\mathcal{A}(c)}(S)(\varepsilon_{S(X)}(z)) & \\ & = & \kappa_i \mathsf{r}_{\mathcal{A}(c)}(S_i)(\mathcal{A}(c)(\kappa_i)^{-1}(\varepsilon_{S(X)}(z))) & \text{iff} \quad \neg\mathcal{A}(c)(\kappa_i)(\bot) \in \varepsilon_{S(X)}(z) \\ & = & \kappa_i \mathsf{r}_{\mathcal{A}(c)}(S_i)(\varepsilon_{S_i(X)}(u)) & \text{iff} \quad z = \kappa_i(u) \\ & \overset{\text{(IH)}}{=} & \kappa_i S_i(\varepsilon_X(u)) & \text{iff} \quad z = \kappa_i(u) \\ & = & S(\varepsilon_X)(z). & \end{array}$$

- In case $S = \mathcal{P}_{\text{fin}} S_1$ we first note that for $z \in \mathcal{P}_{\text{fin}}(S(X))$,

$$\mathcal{A}(c)(\mathcal{P}_{\text{fin}})^{-1}(\varepsilon_{S(X)}(z)) = \{\alpha \in S(X) \mid z \in \alpha^{\mathcal{P}_{\text{fin}}}\} = \{\alpha \in S(X) \mid z \subseteq \alpha\}.$$

Therefore,

$$\begin{array}{rcl}
\mathsf{r}_{\mathcal{A}(c)}(S)(\varepsilon_{S(X)}(z)) & = & \{\mathsf{r}_{\mathcal{A}(c)}(S_1)(V) \mid \mathcal{A}(c)(\mathcal{P}_{\mathrm{fin}})^{-1}(\varepsilon_{S(X)}(z)) \subseteq V\} \\
& = & \{\mathsf{r}_{\mathcal{A}(c)}(S_1)(V) \mid \{\alpha \in S(X) \mid z \subseteq \alpha\} \subseteq V\} \\
& \overset{(*)}{=} & \{S_1(\varepsilon_X)(u) \mid u \in z\} \\
& = & S(\varepsilon_X)(z).
\end{array}$$

where the equation $\overset{(*)}{=}$ is obtained as follows.

($\supseteq$) For $u \in z$ take $V = \varepsilon_{S_1(X)}(u)$, so that $S_1(\varepsilon_X)(u) = \mathsf{r}_{\mathcal{A}(c)}(S_1)(\varepsilon_{S_1(X)}(u)) = \mathsf{r}_{\mathcal{A}(c)}(S_1)(V)$ by (IH). What remains is $\{\alpha \mid z \subseteq \alpha\} \subseteq V$. But this is obvious.

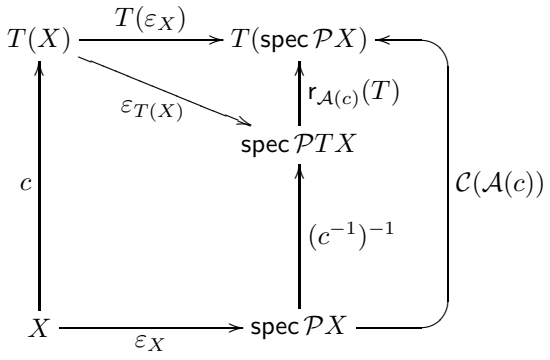
($\subseteq$) Because we have a finite powerset we may assume that z is of the form $\{u_1, \ldots, u_n\}$. Let V be a given ultrafilter with $\{\alpha \mid z \subseteq \alpha\} \subseteq V$. Then $z = \{u_1\} \cup \cdots \cup \{u_n\} \in V$, so that $\{u_i\} \in V$ for a (unique) i. Obviously, $\varepsilon_{S_1(X)}(u_i) \subseteq V$, but also the reverse inclusion holds: if $\alpha \in V$, then $\alpha \cap \{u_i\} \in V$, so that $u_i \in \alpha$. Now we are done by (IH).

□

Theorem 5.7. *Let $c\colon X \to T(X)$ be a coalgebra of a finite KPF. The map $\varepsilon_X\colon X \to \mathsf{spec}\mathcal{P}X$, given by $x \mapsto \{\alpha \mid x \in \alpha\}$, is then a homomorphism of coalgebras from c to the ultrafilter extension $\mathcal{C}(\mathcal{A}(c))$.*

Moreover, c and $\mathcal{C}(\mathcal{A}(c))$ satisfy the same state formulas of the logic $MSML_T$.

Proof. Consider the following diagram.



The triangle at the top commutes by the previous lemma. The lower left square obviously commutes.

For a state formula $\varphi \in \mathcal{L}_{\mathrm{Id}}$, we have:

$$\begin{array}{rcll} [\![\,\varphi\,]\!]^{\mathcal{C}(\mathcal{A}(c))} = \top & \iff & \forall U \in \mathsf{spec}\,\mathcal{P}X.\, \mathsf{r}_{\mathcal{A}(c)}(\mathrm{Id})(U) = U \in [\![\,\varphi\,]\!]^{\mathcal{C}(\mathcal{A}(c))} & \\ & \iff & \forall U \in \mathsf{spec}\,\mathcal{P}X.\, [\![\,\varphi\,]\!]^{\mathcal{A}(c)} \in U & \text{by Lemma 5.4} \\ & \iff & [\![\,\varphi\,]\!]^{\mathcal{A}(c)} = \top & \text{by Corollary 2.12} \\ & \iff & [\![\,\varphi\,]\!]^{c} = \top & \text{see Example 4.6.} \end{array}$$

□

5.3. Final coalgebras

In this final part we show how the canonical model coalgebra $\mathcal{C}(\mathcal{L}_T)$ constructed out of the Lindenbaum model $\mathcal{L}_T$ from Example 4.4 is final, and use this fact to give a new proof of the Hennessey–Milner type characterisation result of [23, 24] for coalgebras. A similar final coalgebra, constructed purely syntactically, appears in [24] — but only for polynomial functors without powerset.

Theorem 5.8. *For a finite KPF T, the "canonical model" coalgebra $\mathcal{C}(\mathcal{L}_T)$ is final in the category $\boldsymbol{CoAlg}(T)$. For an arbitrary T-coalgebra $X \xrightarrow{c} T(X)$, the unique homomorphism $! : X \to \mathsf{spec}\,\mathcal{L}_T(Id)$ is given by*

$$! \;=\; \mathcal{C}([\![-]\!]) \circ \varepsilon_X \;=\; \lambda x \in X.\, \{|\varphi| \mid \varphi \in \mathcal{L}_{Id} \text{ with } x \in [\![\,\varphi\,]\!]\}$$

where $[\![-]\!]$ is the interpretation homomorphism $\mathcal{L}_T \to \mathcal{A}(c)$ of MBAOs from Proposition 4.8, and ε_X is the ultrafilter extension map from Theorem 5.7.

Proof. By Theorem 5.7 we know that ! is a homomorphism, and so we only have to prove its uniqueness. So suppose $f\colon X \to \mathsf{spec}\,\mathcal{L}_T(\mathrm{Id})$ is a homomorphism, *i.e.* satisfies $\mathsf{r}_{\mathcal{L}_T}(T) \circ \mathsf{next}^{-1} \circ f = T(f) \circ c$. What we have to prove is $f(x) = !(x)$, *i.e.*:

$$|\varphi| \in f(x) \iff x \in [\![\,\varphi\,]\!].$$

We shall used induction on the structure of $\varphi \in \mathcal{L}_{\mathrm{Id}}$:

- The case where $\varphi = \bot$ is obvious, because $f(x)$ is an ultrafilter.
- Similarly, if $\varphi = \varphi_1 \to \varphi_2$ we have:

$$\begin{array}{rcl} |\varphi| = |\varphi_1| \to |\varphi_2| \in f(x) & \iff & \big(|\varphi_1| \in f(x) \text{ implies } |\varphi_2| \in f(x)\big) \\ & \overset{\text{(IH)}}{\iff} & \big(x \in [\![\,\varphi_1\,]\!] \text{ implies } x \in [\![\,\varphi_2\,]\!]\big) \\ & \iff & x \in [\![\,\varphi_1 \to \varphi_2\,]\!]. \end{array}$$

- The final case $\varphi = \mathsf{next}\psi$, for $\psi \in \mathsf{Form}_T$ is most interesting:

$$\begin{array}{lll} x \in [\![\varphi]\!] = [\![\mathsf{next}\psi]\!] = c^{-1}([\![\psi]\!]) & & \\ \iff & c(x) \in [\![\psi]\!] = T(f)^{-1}\big([\![\psi]\!]^{\mathcal{C}(\mathcal{L}_T)}\big) & \text{by Corollary 3.8} \\ \iff & T(f)(c(x)) = \mathsf{r}_{\mathcal{L}_T}(T)(\mathsf{next}^{-1}(f(x))) \in [\![\psi]\!]^{\mathcal{C}(\mathcal{L}_T)} & \text{by assumption} \\ \iff & [\![\psi]\!]^{\mathcal{L}_T} = |\psi| \in \mathsf{next}^{-1}(f(x)) & \text{by Lemma 5.4} \\ \iff & |\varphi| = |\mathsf{next}\psi| \in f(x). & \end{array}$$

□

Using that the canonical model is final, we get a version of a Hennessey–Milner style result (see [6]) for coalgebras. It is the same as [23] (Prop. 4.8), but with an easier proof using final coalgebras.

Corollary 5.9. *Let $X \xrightarrow{c} T(X)$ and $Y \xrightarrow{d} T(Y)$ be two coalgebras of a finite KPF T. Two states $x \in X$ and $y \in Y$ are then bisimilar if and only if they satisfy the same state formulas.*

Proof. Proposition 2.9 says that x, y are bisimilar if and only $!(x) = !(y)$. So the result follows from the description of the unique map $!$ to the final coalgebra $\mathcal{C}(\mathcal{L}_T)$ from the previous theorem. □

6. Conclusion

This paper extends the semantical approach of [12] from single-sorted to many-sorted modal logic, and creates a setting in which several earlier developments and results (notably from [23]) are suitably generalised (or adapted) so that they can find their natural place.

Acknowledgements. Thanks are due to Martin Rößiger for explaining and discussing various aspects of his work.

References

[1] A. Baltag, A logic for coalgebraic simulation, edited by H. Reichel, *Coalgebraic Methods in Computer Science*. Elsevier, Amsterdam, *Electon. Notes Theor. Comput. Sci.* **33** (2000).
[2] B.A. Davey and H.A. Priestley, *Introduction to Lattices and Order*, Math. Textbooks. Cambridge Univ. Press (1990).
[3] R. Goldblatt, *Mathematics of Modality*. Stanford, *CSLI Lecture Notes* **43** (1993).
[4] R. Goldblatt, Duality for some categories of coalgebras. *Algebra Universalis* (to appear).
[5] R. Goldblatt, What is the coalgebraic analogue of Birkhoff's variety theorem? *Theoret. Comput. Sci.* (to appear).
[6] M. Hennessy and R. Milner, On observing non-determinism and concurrency, edited by J.W. de Bakker and J. van Leeuwen, *Mathematical Foundations of Computer Science*. Springer, Berlin, *Lecture Notes in Comput. Sci.* **85** (1980) 299-309.

[7] U. Hensel, M. Huisman, B. Jacobs, and H. Tews, Reasoning about classes in object-oriented languages: Logical models and tools, edited by Ch. Hankin, *European Symposium on Programming*. Springer, Berlin, *Lecture Notes in Comput. Sci.* **1381** (1998) 105-121.
[8] C. Hermida and B. Jacobs, Structural induction and coinduction in a fibrational setting. *Inform. and Comput.* **145** (1998) 107-152.
[9] B. Jacobs, Objects and classes, co-algebraically, edited by B. Freitag, C.B. Jones, C. Lengauer and H.-J. Schek, *Object-Orientation with Parallelism and Persistence*. Kluwer Acad. Publ. (1996) 83-103.
[10] B. Jacobs, *Categorical Logic and Type Theory*. North Holland, Amsterdam (1999).
[11] B. Jacobs, *The temporal logic of coalgebras via Galois algebras*, Technical Report CSI-R9906. Comput. Sci. Inst. Univ. of Nijmegen, *Math. Structures Comput. Sci.* (to appear).
[12] B. Jacobs, Towards a duality result in coalgebraic modal logic, edited by H. Reichel, *Coalgebraic Methods in Computer Science*. Elsevier, Amsterdam, *Electon. Notes Theor. Comput. Sci.* **33** (2000).
[13] B. Jacobs and J. Rutten, A tutorial on (co)algebras and (co)induction. *EATCS Bull.* **62** (1997) 222-259.
[14] B. Jacobs, J. van den Berg, M. Huisman, M. van Berkum, U. Hensel and H. Tews, Reasoning about classes in Java (preliminary report), in *Object-Oriented Programming, Systems, Languages and Applications (OOPSLA)*. ACM Press (1998) 329-340.
[15] P.T. Johnstone, *Stone Spaces*. Cambridge Univ. Press, *Cambridge Stud. Adv. Math.* **3** (1982).
[16] B. Jónsson and A. Tarski, Boolean algebras with operators I. *Amer. J. Math.* **73** (1951) 891-939.
[17] A. Kurz, Specifying coalgebras with modal logic. *Theoret. Comput. Sci.* (to appear).
[18] F.W. Lawvere, Functorial semantics. *Proc. Nat. Acad. Sci. U.S.A.* **50** (1963) 869-872.
[19] E.J. Lemmon, Algebraic semantics for modal logics II. *J. Symbolic Logic* **31** (1966) 191-218.
[20] L.S. Moss, Coalgebraic logic. *Ann. Pure Appl. Logic* **96** (1999) 277-317; *Erratum* in *Ann. Pure Appl. Logic* **99** (1999) 241-259.
[21] H. Reichel, An approach to object semantics based on terminal co-algebras. *Math. Structures Comput. Sci.* **5** (1995) 129-152.
[22] M. Rößiger, Languages for coalgebras on datafunctors, edited by B. Jacobs and J. Rutten, *Coalgebraic Methods in Computer Science*. Elsevier, Amsterdam, *Electron. Notes Theor. Comput. Sci.* **19** (1999).
[23] M. Rößiger, Coalgebras and modal logic, edited by H. Reichel, *Coalgebraic Methods in Computer Science*. Elsevier, Amsterdam, *Electron. Notes Theor. Comput. Sci.* **33** (2000).
[24] M. Rößiger, From modal logic to terminal coalgebras. *Theoret. Comput. Sci.* (to appear).
[25] J. Rothe, H. Tews and B. Jacobs, The coalgebraic class specification language CCSL. *J. Universal Comput. Sci.* **7** (2001).
[26] J. Rutten, Universal coalgebra: A theory of systems. *Theoret. Comput. Sci.* **249** (2000) 3-80.
[27] M.H. Stone, The theory of representations for Boolean algebra. *Trans. Amer. Math. Soc.* **40** (1936) 37-111.
[28] M.H. Stone, Applications of the theory of Boolean rings to general topology. *Trans. Amer. Math. Soc.* **41** (1937) 375-481.
[29] D. Turi and J. Rutten, On the foundations of final semantics: non-standard sets, metric spaces and partial orders. *Math. Structures Comput. Sci.* **8** (1998) 481-540.
[30] Y. Venema, Points, lines and diamonds: a two-sorted modal logic for projective planes. *J. Logic Comput.* **9** (1999) 601-621.
[31] S. Vickers, *Topology Via Logic*. Cambridge Univ. Press, *Tracts Theor. Comput. Sci.* **5** (1989).

Accepted March 9, 2001.