Modal Logics for Coalgebras – A Survey

Ichiro HASUO[*]
Department of Mathematical and Computing Sciences,
Tokyo Institute of Technology.
`hasuo2@is.titech.ac.jp`, `http://www.is.titech.ac.jp/~hasuo2`

August 2003

Abstract

The notion of coalgebra is an abstraction of state-based systems such as automata, labelled transition systems and Kripke models. It has proved that in order to state about behaviors of coalgebras, some kinds of modal logics are useful. This survey is to summarize several different ways of defining such modal languages.

1 Introduction

The notion of coalgebra is an abstraction of *state-based systems*, such as automata, labelled transition systems and Kripke models. We can give some properties common in them:

- Behaviors of the system depends on its internal state which is, however, invisible to the user (the *black-box* view).
- The system is reactive, not necessarily terminating, and interacts with its environment.
- The system comes with a set of operations, through which this interaction takes place.

The notion of coalgebras offers a general theory to consider a large class of such systems.

What is also interesting about coalgebras is that the notion is the dual of the well-known structure *algebra*, hence many results and insights in the well-studied area of *universal algebra* can be applied to the theory of coalgebras.

To state about algebras, we usually make usage of equational logic. For coalgebras it has proved that some kinds of modal languages are useful,[1] and several authors have introduced different ways to obtain modal languages for coalgebras. This survey is to summarize those approaches.

In the second section, we make a brief review on basic notions around coalgebras, accompanied with some examples of coalgebras. The third section is an introduction to modal logics for coalgebras. There we do not go into concrete modal languages, but we will see what modal logics for coalgebras are all about, and two desirable properties of modal logics for coalgebras, namely *adequacy* and *expressivity*, and two desirable properties of proof systems for those logics, namely *soundness* and *completeness*, are introduced. In the remaining sections we summarize some authors' approaches to modal logics for coalgebras and compare them with one another.

There are a number of research topics involving coalgebras; introductions and pointers to literatures can be found in [19], [10] and [15].

[*]This report was written during the author's stay at Laboratory for Verification and Semantics, National Institute of Advanced Industrial Science and Technology, Japan (AIST).

[1]"Modal logic is to coalgebras what equational logic is to algebras" [10].

2 Coalgebras and other basic notions

The materials in this section are mostly from [4], which is based on [15].

2.1 Coalgebras and morphisms

Definition 2.1 (Coalgebras) Let $\mathbb{C}$ be a category, $T : \mathbb{C} \to \mathbb{C}$ an endofunctor on $\mathbb{C}$. A *T-coalgebra* is a pair of a $\mathbb{C}$-object C and a $\mathbb{C}$-arrow $\gamma : C \to TC$, i.e. (C, γ). T is said to be its *signature functor* or *type*, C its *carrier*, $\mathbb{C}$ its *base category* and γ its *transition map* or *coalgebraic structure*. ■

In this paper our focus is set on coalgebras whose base category is $\mathbb{S}\mathrm{et}$. Later we will make usage of some properties unique to $\mathbb{S}\mathrm{et}$, such as another characterization of accessibility of functors.

Definition 2.2 (Morphisms of coalgebras) Let (C, γ) and (D, δ) be T-coalgebras, and $f : C \to D$ a $\mathbb{C}$-arrow. f is said to be a *morphism of T-coalgebras* or *T-morphism*, if $\delta \circ f = Tf \circ \gamma$, i.e. the following diagram of $\mathbb{C}$ commutes.

$$\begin{tikzcd} C \arrow[r, "f"] \arrow[d, "\gamma"'] & D \arrow[d, "\delta"] \\ TC \arrow[r, "Tf"'] & TD \end{tikzcd}$$

We see by the functoriality of T that:

- an identity arrow is a morphism;
- composition of two morphisms is again a morphism.

Hence T-coalgebras and T-morphisms again forms a category, which we denote by $\mathbb{C}\mathrm{o}\mathbb{A}\mathrm{lg}(T)$. ■

Example 2.3 (Stream Automata) A *stream automaton* is a very simple machine, with only a button and a display. One letter is indicated on the display every time the button is pressed down.

Such a class of machines can be formalized as follows, at first sight. Let L be a (fixed) set of letters that can be on the display. Then a stream automaton is nothing but a triple[2]

$$\langle S,\ \mathsf{hd} : S \to L,\ \mathsf{tl} : S \to S \rangle$$

where S is a nonempty set of its internal states, and on pressing the button with the machine's state $s \in S$, we see $\mathsf{hd}(s)$ on the display, and the state now changes into $\mathsf{tl}(s)$.

$$\begin{tikzcd} & S \arrow[dl, "\mathsf{hd}"'] \arrow[d, dashed, "\langle \mathsf{hd}, \mathsf{tl} \rangle"] \arrow[dr, "\mathsf{tl}"] & \\ L & L \times S \arrow[l, "\mathrm{pr}_1"] \arrow[r, "\mathrm{pr}_2"'] & S \end{tikzcd}$$

By taking a Cartesian product, we can now regard this as a coalgebra

$$(S,\ \langle \mathsf{hd}, \mathsf{tl} \rangle : S \to L \times S)$$

[2] hd is for "head" and tl is for "tail", as an observation later shows.

for the signature functor[3]

$$TX := L \times X,$$
$$Tf := L \times f = \mathrm{id}_L \times f.$$

Again at first sight we can define a *morphism* of stream automata as follows:

Let $\langle S, \mathsf{hd}, \mathsf{tl}\rangle$ and $\langle S', \mathsf{hd}, \mathsf{tl}\rangle$ be stream automata with a common set of outputs L, and $f : S \to S'$. f is a *morphism* if for every $s \in S$,

$$\mathsf{hd}(s) = \mathsf{hd}' \circ f(s),$$
$$f \circ \mathsf{tl}(s) = \mathsf{tl}' \circ f(s).$$

This exactly coincides with the definition of morphisms of coalgebras, i.e.

$$\langle \mathsf{hd}', \mathsf{tl}'\rangle \circ f(s) = (\mathrm{id}_L \times f) \circ \langle \mathsf{hd}, \mathsf{tl}\rangle(s).$$

∎

Example 2.4 (Labelled transition systems) A *labeled transition system* with the set of actions (or labels) L is formalized as a pair of the set of states S and the three-place relation $R \subseteq S \times L \times S$, i.e. $\langle S, R\rangle$. This can be rewritten as $\langle S, (\xrightarrow{l})_{l \in L}\rangle$, where $\xrightarrow{l}$ is a binary relation defined by

$$s \xrightarrow{l} s' \quad \text{iff} \quad (s, l, s') \in R.$$

Now using the trick of 'relation into function', we obtain a coalgebraic formulation of labelled transition system. Define $\sigma : S \to \mathcal{P}(L \times S)$ by

$$\sigma(s) := \{(l, s') \in L \times S \mid (s, l, s') \in R, \text{ i.e. } s \xrightarrow{l} s'\},$$

then a labelled transition system (S, σ) is a coalgebra for the signature functor

$$TX := \mathcal{P}(L \times X),$$
$$Tf := \mathcal{P}(L \times f),$$

where, for $g : A \to B$, $\mathcal{P}g : \mathcal{P}A \to \mathcal{P}B$ is defined by

$$(\mathcal{P}g)(\mathfrak{a}) := g[\mathfrak{a}] = (\text{the direct image of } \mathfrak{a} \text{ by } g).$$

∎

Example 2.5 (Kripke models) A *Kripke model* for a set of atomic formulas A is a triple $\langle S, R, V\rangle$, where S is the set of *possible worlds* (or states), R ($\subseteq S \times S$) is an *accessibility relation*, and $V : A \to \mathcal{P}S$ is a valuation of atomic formulas ($V(a)$ is the set of states where a is true).

Again by making a similar trick as above, we obtain a coalgebraic formalization of Kripke models. For a Kripke model $\langle S, R, V\rangle$, define $\mathsf{next} : S \to \mathcal{P}S$ and $\mathsf{prop} : S \to \mathcal{P}A$ by:

$$\mathsf{next}(s) := \{s' \in S \mid sRs'\} = (\text{states that are possibly the next of } s),$$
$$\mathsf{prop}(s) := \{a \in A \mid s \in V(a)\} = (\text{formulas true at } s).$$

[3]The identity arrow id_A associated to A is often denoted by A itself.

Taking a Cartesian product,

$$(S, \langle \mathsf{next}, \mathsf{prop} \rangle : S \to \mathcal{P}S \times \mathcal{P}A)$$

is a coalgebra for the signature functor

$$TX := \mathcal{P}X \times \mathcal{P}A,$$
$$Tf := \mathcal{P}f \times \mathcal{P}A.$$

Consider a morphism of Kripke models, which are now considered as $T(= \mathcal{P}\,\mathrm{id} \times \mathcal{P}A)$-coalgebras. Let $(S, \langle \mathsf{next}, \mathsf{prop} \rangle) = \langle S, R, V \rangle$ and $(S', \langle \mathsf{next}', \mathsf{prop}' \rangle) = \langle S', R', V' \rangle$ be Kripke models, and $f : S \to S'$. Relaxing the above definition, we obtain that f is a morphism (w.r.t. T-coalgebras) iff for every $s \in S$,

- $\mathsf{prop}(s) = \mathsf{prop}' \circ f(s)$, that is,
  - for every $a \in A$, $s \models_{\langle S,R,V \rangle} a$ iff $f(s) \models_{\langle S',R',V' \rangle} a$, and
- $f[\mathsf{next}(s)] = \mathsf{next}' \circ f(s)$, that is,
  - $s_1 R s_2$ implies $f(s_1) R' f(s_2)$, and
  - if $f(s_1) R' s'_2$, then there exists $s_2 \in S$ such that $s_1 R s_2$ and $f(s_2) = s'_2$,

which is nothing but the condition of *p-morphism* or *pseudo epimorphism* [20]. ∎

2.2 Behavioral equivalence and bisimilarity

As stated in the introduction, the theory of coalgebras is to study such systems whose internal states we users cannot see, but whose outputs (or *behaviors*) we can see. Hence we are concerned with not a state itself, but a class of states which share the same behaviors, i.e. an equivalence class modulo a certain relation.

There are two candidates for the relation: *behavioral equivalence* [9] and *bisimilarity*. It is noted again that the base category is supposed to be $\mathbb{S}\mathrm{et}$ in this paper.

Definition 2.6 (Behavioral equivalence) Let (C, γ) and (D, δ) be T-coalgebras, $c \in C$ and $d \in D$. c and d are *behaviorally equivalent*, which is denoted by $c \approx d$, if there exist a T-coalgebra (E, ϵ) and T-morphisms $f : (C, \gamma) \to (E, \epsilon)$, $g : (D, \delta) \to (E, \epsilon)$ such that $f(c) = g(d)$.

$$\begin{tikzcd}
(C,\gamma) \arrow[dr, "f"] & & (D,\delta) \arrow[dl, "g"'] \\
& (E,\epsilon) &
\end{tikzcd}$$

∎

Definition 2.7 (Bisimilarity) Let (C, γ) and (D, δ) be T-coalgebras, and $B \subseteq C \times D$ a binary relation on C and D. B is said to be a *bisimulation* if there exists a transition map $\beta : B \to TB$ which makes the projections $\mathrm{pr}_1 : B \to C$ and $\mathrm{pr}_2 : B \to D$ both T-morphisms.

$$\begin{tikzcd}
& (B,\beta) \arrow[dl, "\mathrm{pr}_1"'] \arrow[dr, "\mathrm{pr}_2"] & \\
(C,\gamma) & & (D,\delta)
\end{tikzcd}$$

$c \in C$ and $d \in D$ are *bisimilar* if for some bisimulation B, cBd. ∎

It is not trivial that behavioral equivalence is an equivalence relation in fact; the question is in its transitivity. It is easily shown that if the base category has pushouts then so does the category of T-coalgebras and morphisms (the forgetful functor preserves pushouts, moreover). This is the case here since $\mathbb{S}\mathrm{et}$ has pushouts. Then the transitivity of behavioral equivalence is easily established using pushouts.[4]

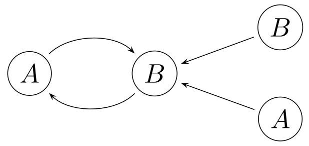
Having established that behavioral equivalence is actually an equivalence relation, it is easy that bisimilarity yields behavioral equivalence. Moreover, if the signature functor T *preserves weak pullbacks* [19], then the converse also holds, i.e. behavioral equivalence exactly coincides with bisimilarity.[5]

The definition of behavioral equivalence seems natural if we think of a morphism as *behavior-preserving map*, and since it is the more general notion[6] we will mainly work with it.

Example 2.8 (Behavioral equivalence of stream automata) Given a stream automaton $(S, \langle \mathsf{hd}, \mathsf{tl} \rangle)$ and a state $s \in S$, it seems obvious that the *behavior* of s is the stream of letters

$$(\mathsf{hd}(s),\ \mathsf{hd} \circ \mathsf{tl}(s),\ \mathsf{hd} \circ \mathsf{tl} \circ \mathsf{tl}(s),\ \dots)$$

which will appear in pressing the button repeatedly. Hence we obtain a mapping of a state into its behavior, $\mathsf{beh}_{(S, \langle \mathsf{hd}, \mathsf{tl} \rangle)} : S \to L^\omega$, given by $\mathsf{beh}_{(S, \langle \mathsf{hd}, \mathsf{tl} \rangle)}(s) := (\mathsf{hd} \circ \mathsf{tl}^n(s))_{n \in \omega}$, where $\mathsf{tl}^n(s) := \mathsf{tl} \circ \mathsf{tl}^{n-1}(s)$ is defined inductively.



The above figure designates one stream automaton, each circle being a state, and each arrow designating transition. It is easy to see that both states labelled with A are behaviorally equivalent, with theie behaviors $A, B, A, B, \dots$. On the other hand, those labelled with B are not.

Now it is easy to verify that

> If $f : (S, \langle \mathsf{hd}, \mathsf{tl} \rangle) \to (S', \langle \mathsf{hd}', \mathsf{tl}' \rangle)$ is a morphism of stream automata, then $\mathsf{beh}_{(S, \langle \mathsf{hd}, \mathsf{tl} \rangle)}(s) = \mathsf{beh}_{(S', \langle \mathsf{hd}', \mathsf{tl}' \rangle)} \circ f(s)$.

Hence if two states $s \in (S, \langle \mathsf{hd}, \mathsf{tl} \rangle)$ and $s' \in (S', \langle \mathsf{hd}', \mathsf{tl}' \rangle)$ are behaviorally equivalent in the sense defined above ($s \approx s'$), then $\mathsf{beh}_{(S, \langle \mathsf{hd}, \mathsf{tl} \rangle)}(s) = \mathsf{beh}_{(S', \langle \mathsf{hd}', \mathsf{tl}' \rangle)}(s')$.

Here is another easy observasion; the set of all streams over the alphabet L forms a stream automaton. Defining

$$\begin{aligned} \mathsf{hd}^\omega((a_n)_{n \in \omega}) &:= a_0, \text{ and} \\ \mathsf{tl}^\omega((a_n)_{n \in \omega}) &:= (a_{n+1})_{n \in \omega}, \end{aligned}$$

$(L^\omega, \langle \mathsf{hd}^\omega, \mathsf{tl}^\omega \rangle)$ is a stream automaton. Moreover, for every stream automaton $(S, \mathsf{hd}, \mathsf{tl})$, $\mathsf{beh} : (S, \mathsf{hd}, \mathsf{tl}) \to (L^\omega, \langle \mathsf{hd}^\omega, \mathsf{tl}^\omega \rangle)$ is a morphism. Hence if two states $s \in (S, \langle \mathsf{hd}, \mathsf{tl} \rangle)$ and $s' \in (S', \langle \mathsf{hd}', \mathsf{tl}' \rangle)$ share the same behavior in common in the sense that $\mathsf{beh}_{(S, \langle \mathsf{hd}, \mathsf{tl} \rangle)}(s) = \mathsf{beh}_{(S', \langle \mathsf{hd}', \mathsf{tl}' \rangle)}(s')$, then $s \approx s'$, with $(L^\omega, \langle \mathsf{hd}^\omega, \mathsf{tl}^\omega \rangle)$ their confluence.

The two observations establish that $s \approx s'$ iff $\mathsf{beh}_{(S, \langle \mathsf{hd}, \mathsf{tl} \rangle)}(s) = \mathsf{beh}_{(S', \langle \mathsf{hd}', \mathsf{tl}' \rangle)}(s')$. ∎

Example 2.9 (Behavioral equivalence in Kripke models) For Kripke models which are thought of as $(\mathcal{P}\,\mathrm{id} \times \mathcal{P}A)$-coalgebras, behavioral equivalence coincides with bisimilarity which is defined above, and moreover with bisimilarity in its *original* form introduced in [21]. For the proof see [15]. ∎

[4]See e.g. [4].

[5]The proof is easy, taking a pullback as a bisimulation.

[6]Examples where behavioral equivalence is in fact the more appropriate is given in [9].

2.3 Simplicity, extensivity and finality

Definition 2.10 (Simple coalgebra, extensive coalgebra) A T-coalgebra (C, γ) is *simple* if for all $c, c' \in C$, $c \approx c'$ implies $c = c'$. It is also said to *satisfy the coinduction proof principle*, in the sense that to show that two states are equal, we have only to show that they are behaviorally equivalent. A simple coalgebra is one without any redundant internal states.

(C, γ) is *extensive* if for every T-coalgebra (D, δ) and $d \in D$, there exists $c \in C$ such that $c \approx d$. We can say that an extensive coalgebra has every possible behavior (as to T) in it. ∎

A coalgebra which is both simple and extensive can be thought of as a system consisting of *all possible behaviors*. What is interesting is that this property can be stated in purely category-theoretical terms.

Definition 2.11 (Final coalgebra) (C, γ) is *final* if it is the final (or terminal) object in the category $\mathbb{C}\mathrm{o}\mathbb{A}\mathrm{lg}(T)$, i.e. for each T-coalgebra (D, δ) there exists exactly one morphism from (D, δ) to (C, γ).[7] ∎

The proofs for claims in this subsection can be found in [4].

Proposition 2.12 *A coalgebra (Z, ζ) is both simple and extensive, if and only if it is final.*

Example 2.13 (Stream automata) For stream automata, i.e. T-coalgebras with $T = L \times \mathrm{id}$, $(L^\omega, \langle \mathsf{hd}^\omega, \mathsf{tl}^\omega \rangle)$ is the final coalgebra. ∎

Theorem 2.14 (Lambek's lemma) *For the final T-coalgebra (Z, ζ), $\zeta : Z \to TZ$ is a bijection.*

In the above we have not mentioned whether the final coalgebra for a signature functor T actually exists or not; the theorem gives a negative example.

Corollary 2.15 *There is no final $\mathcal{P}$-coalgebra, where $\mathcal{P}$ is the powerset functor.*

2.4 Terminal sequence

Under some condition on the signature functor, we can establish the existence of the final T-coalgebra. For the proof we use the notion of *terminal sequence*, whose α-th object $T^\alpha 1$ designates the set of all behaviors that can appear in α-step transitions. Moreover, it also gives rise to the proof method called *induction along the terminal sequence*, which we will use again and again later.

Definition 2.16 (Terminal sequence)

$$\begin{tikzcd} 1 & T1 \arrow[l, "!"'] & T^2 1 \arrow[l, "T!"'] & \cdots \arrow[l, "T^2!"'] & T^\alpha 1 \arrow[l, "T^{\alpha-1}!"'] & \cdots \arrow[l, "T^\alpha !"'] \end{tikzcd}$$

Let T be an endofunctor on $\mathbb{S}\mathrm{et}$. The *terminal sequence* associated with T is a transfinitely long sequential diagram in $\mathbb{S}\mathrm{et}$, consisting of $\mathbb{S}\mathrm{et}$-objects $T^\alpha 1$ for every ordinal α and $\mathbb{S}\mathrm{et}$-arrows $\zeta^\alpha_\beta : T^\alpha 1 \to T^\beta 1$ for all ordinals $\beta \leq \alpha$, satisfying

- $\zeta^\beta_\gamma \circ \zeta^\alpha_\beta = \zeta^\alpha_\gamma$ for all $\gamma \leq \beta \leq \alpha$, and
- $\zeta^{\alpha+1}_{\beta+1} = T\zeta^\alpha_\beta$ for all $\beta \leq \alpha$.

∎

[7]Of course, the final coalgebra is unique up to isomorphism.

Intuitively ζ^α_β ($\beta \leq \alpha$) maps an α-step behavior to its prefixing β-step behavior.

We can construct the terminal sequence for every endofunctor T; the detailed construction is described in [4].

Proposition 2.17 *A T-coalgebra (C, γ) induces a cone $(C, (\gamma_\alpha : C \to T^\alpha 1)_{\alpha \in \mathbb{O}\mathrm{rd}})$ over the terminal sequence associated with T.*

The map $\gamma_\alpha : C \to T^\alpha 1$ maps a state $c \in C$ to its behavior within α-step transitions.

Example 2.18 (Stream automata) For stream automata over an alphabet L, which is thought of as $(L \times \mathrm{id})$-coalgebras, the terminal sequence associated with $L \times \mathrm{id}$ is as follows:

$$\begin{tikzcd} 1 & L \arrow[l, "!"'] & L^2 \arrow[l, "L\times !"'] & \cdots \arrow[l, "L^2\times !"'] & L^n \arrow[l, "L^{n-1}\times !"'] & \cdots \arrow[l, "L^n\times !"'] \end{tikzcd}$$

where the arrow ζ^n_m ($m \leq n$) maps an n-length word over L to its m-length prefix. Given a stream automaton $(C, \gamma = \langle \mathsf{hd}, \mathsf{tl} \rangle)$, it induces a cone $(C, (\gamma_\alpha : C \to T^\alpha 1)_{\alpha \in \mathbb{O}\mathrm{rd}})$. Here the arrow $\gamma_n : C \to T^n 1 \, (= L^n)$ maps $c \in C$ to $(\mathsf{hd}(c), \mathsf{hd} \circ \mathsf{tl}(c), \ldots, \mathsf{hd} \circ \mathsf{tl}^{n-1}(c))$. ∎

Using the terminal sequence, we can establish the existence result of the final coalgebra, if the type T is *accessible*. The notion of accessibility is usually defined in terms of filtered colimits; however, since our focus here is on endofunctors on $\mathbb{S}\mathrm{et}$, we can go another way.

Definition 2.19 (Accessibility) Let κ a regular cardinal, and $T : \mathbb{S}\mathrm{et} \to \mathbb{S}\mathrm{et}$. T is said to be *κ-accessible* if:

> for every set X and every $t \in TX$, there exists $\mathfrak{x}_t \subseteq X$ such that $\#\mathfrak{x}_t < \kappa$ and $t \in (Ti)[T\mathfrak{x}_t]$,[8] where $i : \mathfrak{x}_t \to X$ is the inclusion map.

T is *accessible* if it is κ-accessible for some κ. ∎

Intuitively, a functor is accessible if its action on a large set can be determined by that on small subsets.

Theorem 2.20 (Existence of final coalgebra) *If T is accessible, then there exists the final T-coalgebra.*

Proof. The key observation is as follows. If T is κ-accessible, then the arrow $\zeta^{\kappa+1}_\kappa : T^{\kappa+1}1 \to T^\kappa 1$ in the terminal sequence is monic. Taking its left inverse $\zeta : T^\kappa 1 \to T^{\kappa+1} 1 \, (= T(T^\kappa 1))$, we obtain the final coalgebra $(T^\kappa 1, \zeta)$. Again the detailed proof is found in [4]. ∎

We can extract a proof principle from the proof of the above theorem:

Theorem 2.21 (Induction along the terminal sequence) *Let $T : \mathbb{S}\mathrm{et} \to \mathbb{S}\mathrm{et}$ be a κ-accessible functor, (C, γ), (D, δ) T-coalgebras, $c \in C$ and $d \in D$. Then the following are equivalent:*

1. *$c \approx d$;*
2. *$!_C(c) = !_D(d)$, where $!_C : (C, \gamma) \to (Z, \zeta)$ is the morphism to the final coalgebra uniquely determined by the finality, and so is $!_D$;*
3. *for all $\alpha < \kappa$, $\gamma_\alpha(c) = \delta_\alpha(d)$, where $(C, (\gamma_\alpha : C \to T^\alpha 1)_{\alpha \in \mathbb{O}\mathrm{rd}})$ is a cone over the terminal sequence induced by the T-coalgebra (C, γ), and so is $(D, (\delta_\alpha : D \to T^\alpha 1)_{\alpha \in \mathbb{O}\mathrm{rd}})$.*

The proof of Theorem 2.20 implies that for κ-accessible type T only behaviors within κ steps make sense; the condition 3. says that c and d have the same behavior in common within κ steps. If we are to establish $c \approx d$, then it suffices to show that $\gamma_\alpha(c) = \delta_\alpha(d)$ for all $\alpha < \kappa$ by transfinite induction.

[8]Note that $f[-]$ denotes a direct image.

3 Modal logics for coalgebras

Modal languages have proved to be useful for specification languages for behaviors of coalgebras. This idea easily comes to our mind if we consider coalgebras as a generalization of Kripke models. In this section we describe what modal logics for coalgebras are all about, and some properties which a modal language, or a proof system for it, is expected to possess. Here we do not stick on one certain language.

Let $\mathcal{L}$ be a modal language for T-coalgebras, which is the collection of modal formulas. Several of those will be introduced in the next section.[9] A modal formula in $\mathcal{L}$ is interpreted in a T-coalgebra; that is, given a T-coalgebra (C, γ) and a modal formula ϕ of $\mathcal{L}$, we define the *semantics* of ϕ in (C, γ) as a subset of C, denoted by $[\![\phi]\!]_{(C,\gamma)}$, which is the collection of states of C where ϕ is true. We put $c \models_{(C,\gamma)} \phi$ for $c \in [\![\phi]\!]_{(C,\gamma)}$. We omit subscripts (C, γ) if no confusion comes about.

Given a modal language (or *syntax*) $\mathcal{L}$ and its interpretation in T-coalgebras, there are two properties which they are expected to enjoy, when we take $\mathcal{L}$ for a specification language for T-coalgebras and moreover we adopt the *black-box* view as to coalgebras, i.e. we are concerned with their *behaviors* rather than their internal states. Many authors call those properties in different ways. Here we will follow the terminology in [15].

Definition 3.1 (Adequacy, expressivity) Suppose that $\mathcal{L}$ be a modal language for T-coalgebras and $[\![\phi]\!]_{(C,\gamma)}$ ($\subseteq C$) be defined for every $\phi \in \mathcal{L}$ and every T-coalgebra (C, γ).

$\mathcal{L}$ is said to be *adequate* if behavioral equivalence yields logical equivalence; that is, for $c \in (C, \gamma)$ and $d \in (D, \delta)$ with $c \approx d$, $c \models_{(C,\gamma)} \phi$ iff $d \models_{(D,\delta)} \phi$ for all $\phi \in \mathcal{L}$.

$\mathcal{L}$ is said to be *expressive* if the converse holds, i.e. every two states which are logically equivalent with respect to $\mathcal{L}$ are behaviorally equivalent. ∎

The adequacy of $\mathcal{L}$ means that $\mathcal{L}$ (the logical equivalence w.r.t. $\mathcal{L}$, to be precise) is *not too fine*, and it does not distinguish behaviorally equivalent states. On the other hand, the expressivity of $\mathcal{L}$ means that $\mathcal{L}$ is *fine enough*, distinguishing the smallest difference in behaviors of coalgebras. We can say that, for $\mathcal{L}$ to be a specification language, $\mathcal{L}$ *must* be adequate, and it is *better* if $\mathcal{L}$ is expressive. The case seen here is like what is seen about soundness and completeness of a proof system.

In most cases the adequacy of $\mathcal{L}$ is shown by means of the following lemma:

Lemma 3.2 *Suppose that the following holds as to a modal language $\mathcal{L}$ for T-coalgebras and its semantics $[\![\ ]\!]$:*

> *Let $f : (C, \gamma) \to (D, \delta)$ be a morphism of T-coalgebras and $\phi \in \mathcal{L}$. Then for every $c \in C$, $c \models_{(C,\gamma)} \phi$ iff $f(c) \models_{(D,\delta)} \phi$; or equivalently, $[\![\phi]\!]_{(C,\gamma)} = f^{-1}([\![\phi]\!]_{(D,\delta)})$.*

Then $\mathcal{L}$ is adequate.

Proof. Let $c \in (C, \gamma)$ and $d \in (D, \delta)$ be behaviorally equivalent, having T-morphisms $f : (C, \gamma) \to (E, \epsilon)$ and $g : (D, \delta) \to (E, \epsilon)$ with $f(c) = g(d)$. Then the assumption yields that $c \models \phi$ iff $f(c) \models \phi$ iff $g(d) \models \phi$ iff $d \models \phi$ for every $\phi \in \mathcal{L}$, hence c and d are logically equivalent. ∎

On the other hand, to prove expressivity the following lemma which makes usage of induction along terminal sequence will often prove to be useful. Let T be a κ-accessible endofunctor on $\mathbb{S}\mathrm{et}$. Consider the terminal sequence associated with T, with $\zeta^\kappa_\alpha : T^\kappa 1 \to T^\alpha 1$ an arrow in it. Here ζ^κ_α can also be considered as an arrow in the cone over the terminal sequence induced by the final T-coalgebra (Z, ζ), in the way of Proposition 2.17 (Note that we can assume $Z = T^\kappa 1$).

Lemma 3.3 *Let T be a κ-accessible endofunctor on $\mathbb{S}\mathrm{et}$, and $\mathcal{L}$ a modal language for T-coalgebras which is adequate. Suppose that for each $\alpha < \kappa$ and $z \in T^\alpha 1$ we have a formula $\phi^\alpha_z \in \mathcal{L}$ which satisfies $[\![\phi^\alpha_z]\!]_{(Z,\zeta)} = (\zeta^\kappa_\alpha)^{-1}(\{z\})$ where (Z, ζ) and ζ^κ_α are as described above. Then $\mathcal{L}$ is expressive.*

[9] In fact, one of those introduced in the next section does not obey the custom of modal languages; an argument of its modal operator is not a formula, but what is obtained by, so to speak, lifting a formula by the type T.

Proof. Take two states $c \in (C, \gamma)$ and $d \in (D, \delta)$ of T-coalgebras which are not behaviorally equivalent. It suffices to give a formula $\phi \in \mathcal{L}$ such that $c \models \phi$ and $d \not\models \phi$. Let $!_{(C,\gamma)} : (C, \gamma) \to (Z, \zeta)$ the unique arrow into the final T-coalgebra and so is $!_{(D,\delta)}$. Then the adequacy of $\mathcal{L}$ yields that c and $!_{(C,\gamma)}(c)$ are logically equivalent and so are d and $!_{(D,\delta)}(d)$. Hence we can assume that c and d are both states in (Z, ζ).

The proof principle of induction along the terminal sequence (Theorem 2.21) shows that $c \not\approx d$ with $c, d \in (Z, \zeta)$ implies the existence of $\alpha\,(< \kappa)$ such that $\zeta^{\kappa}_{\alpha}(c) \neq \zeta^{\kappa}_{\alpha}(d)$. Take $\phi := \phi^{\alpha}_{\zeta^{\kappa}_{\alpha}(c)}$. Then by the assumption $[\![\phi]\!]_{(Z,\zeta)} = (\zeta^{\kappa}_{\alpha})^{-1}(\{\zeta^{\kappa}_{\alpha}(c)\})$, to which c belongs but d does not. ∎

Once we have obtained a modal language $\mathcal{L}$ for T-coalgebras (which must be adequate and is expected to be expressive), we are then concerned with a proof system (which we denote by Π here) for $\mathcal{L}$, which will deduce *only* valid formulas, and preferably *every* valid formulas. The first property is called *soundness*, and the second *completeness*.

Definition 3.4 (Soundness, completeness) A proof system Π is said to be *sound* if every formula ϕ deduced by Π is valid, that is, $[\![\phi]\!]_{(C,\gamma)} = C$ for every T-coalgebra (C, γ). Π is said to be *complete* if every valid formula in $\mathcal{L}$ can be deduced by Π. ∎

We will denote by $\models_{(C,\gamma)} \phi$ the fact that a formula ϕ is valid in a T-coalgebra (C, γ). If ϕ is valid in every T-coalgebra, we will put just $\models_T \phi$.

In the following sections we will introduce some approaches taken by different authors, and summarize them, considering

- the class of $\mathbb{S}\mathrm{et}$-endofunctors which each method can handle;
- how syntax and semantics are given;
- the results on adequacy or expressivity;
- whether a proof system is given or not, and if it is, the results on its soundness or completeness.

The order of appearance is basically in accordance with the historical development. Remark that notations which appear in each section is ad hoc, valid only in that very section.

4 Moss's coalgebraic logic

[12] discusses a generalization of infinirary modal logic to a logic for coalgebras, which he calls *coalgebraic logic*. Here we will follow the line presented in [15] which makes usage of induction along the terminal sequence for the expressivity result.

Its syntax, as is already mentioned, does not follow the custom of standard modal languages in which a modality is a unary operator whose argument is a formula. The language $\mathcal{L}^{\alpha}_{T}$ is parametrized by the signature functor T and the degree of conjunction α.

Definition 4.1 (Syntax) Let T be a $\mathbb{S}\mathrm{et}$-endofunctor and α a regular cardinal. The language $\mathcal{L}^{\alpha}_{T}$ is defined by the following inductive BNF definition:

$$\mathcal{L}^{\alpha}_{T} \ni \phi ::= \bigwedge \Phi \mid \nabla t$$

where $\Phi \subseteq \mathcal{L}^{\alpha}_{T}$, $\#\Phi < \alpha$ and $t \in T(\mathcal{L}^{\alpha}_{T})$. ∎

The definition can be put differently as follows. Consider the α-bounded powerset functor $\mathcal{P}_{\alpha} : \mathbb{S}\mathrm{et} \to \mathbb{S}\mathrm{et}$ defined by:

$$\begin{array}{lclcl} \mathcal{P}_\alpha X & := & \{\mathfrak{x} \subseteq X \mid \#\mathfrak{x} < \alpha\}; & & \\ \mathcal{P}_\alpha f & : & \mathcal{P}_\alpha A & \to & \mathcal{P}_\alpha B \\ & & \mathfrak{a} & \mapsto & f[\mathfrak{a}] \end{array}$$

where $f : A \to B$. And let the functor $L := \mathcal{P}_\alpha + T$. Then $\mathcal{L}_T^\alpha$ is nothing but the carrier set of the initial L-algebra, with its structure map $[\bigwedge, \nabla] : \mathcal{P}_\alpha \mathcal{L}_T^\alpha + T\mathcal{L}_T^\alpha \to \mathcal{L}_T^\alpha$. In this section we are concerned only with an accessible functor T (Definition 2.19), which makes L also accessible and hence yields the existence of the initial L-algebra, i.e. the validity of the above definition. It is notable that the language cannot be even defined in this way for the covariant powerset functor $\mathcal{P}$, which is not accessible.

It is noted that $\mathcal{L}_T^\alpha$ is an infinitary language unless $\alpha = \omega$, and $\top := \bigwedge \emptyset$ is a formula in $\mathcal{L}_T^\alpha$.

It seems to be the key of Moss's approach that the behavior of the next state is designated in terms of the functor application to $\mathcal{L}_T^\alpha$, and in order to give its semantics, we consider extending T and applying it to *relations*, not only to sets or functions.

Definition 4.2 (Relator) The category $\mathbb{R}\mathrm{el}$ is that of sets and relations, i.e. its object is a (small) set and its arrow is a relation between sets. An identity arrow is given by the diagonal relation, i.e.

$$\mathrm{id}_S = \triangle_S = \{(s, s) \mid s \in S\}.$$

$\mathbb{S}\mathrm{et}$ can be embedded into $\mathbb{R}\mathrm{el}$ by taking the graph $\mathcal{G}_f$ of each function f.

A *relator* is an endofunctor on $\mathbb{R}\mathrm{el}$.

Let $T : \mathbb{S}\mathrm{et} \to \mathbb{S}\mathrm{et}$ and $\hat{T} : \mathbb{R}\mathrm{el} \to \mathbb{R}\mathrm{el}$ both functors. A relator $\hat{T}$ is said to be the *extension* of T if $\hat{T}S = TS$ for each small set S and $\hat{T}\mathcal{G}_f = \mathcal{G}_{Tf}$ for each function $f : A \to B$, where $\mathcal{G}_f$ is an $\mathbb{R}\mathrm{el}$-arrow from A to B (not a $\mathbb{R}\mathrm{el}$-object as a subset of $A \times B$). ∎

We use the extention of the signature functor T to a relator in order to give semantics. The following theorem, due to [2], ensures that it is possible:

Theorem 4.3 *Let $T : \mathbb{S}\mathrm{et} \to \mathbb{S}\mathrm{et}$. T can be extended to a relator if and only if T preserves weak pullbacks. In that case, the extension $\hat{T}$ of T is given by:*

$$\hat{T}R = \mathcal{G}_{T\pi_2} \circ (\mathcal{G}_{T\pi_1})^{\mathrm{op}},$$

where $R : A \to B$ is a $\mathbb{R}\mathrm{el}$-arrow, $\pi_1 : R \to A$ and $\pi_2 : R \to B$ are projections (here R is thought of as a subset of $A \times B$). $-^{\mathrm{op}}$ designates the converse of a relation. In other words,

$$\hat{T}R = \{(T\pi_1(t), T\pi_2(t)) \in TA \times TB \mid t \in TR\},$$

where TR is a functor T applied to a $\mathbb{S}\mathrm{et}$-object R.

In fact, the word *relator* is used by different authors in different ways. Another application of the notion of relator in a coalgebraic setting can be found in [18]. The book [1] gives a good insight and examples of applications to algebraic approaches to programming.

Proposition 4.4 *Let T be a $\mathbb{S}\mathrm{et}$-endofunctor which preserves weak pullbacks, hence the extension of T to a relator is given by $\hat{T}(R) = \mathcal{G}_{T\pi_2} \circ (\mathcal{G}_{T\pi_1})^{\mathrm{op}}$ as above. Then $\hat{T}$*

- *preserves converse, i.e. $\hat{T}(R^{\mathrm{op}}) = (\hat{T}R)^{\mathrm{op}}$;*
- *is order-preserving, i.e. if $R \subseteq S$ for $\mathbb{R}\mathrm{el}$-arrows R and S, then $\hat{T}R \subseteq \hat{T}S$.*

The semantics in a T-coalgebra (C, γ) is now given by the universal arrow from the initial $L(= \mathcal{P}_\alpha + T)$-algebra $(\mathcal{L}_T^\alpha, [\bigwedge, \nabla])$ to an L-algebra which is induced by (C, γ) in terms of the extension of T to a relator:

Definition 4.5 (Semantics) Let T be an accessible $\mathbb{S}\mathrm{et}$-endofunctor which preserves weak pullbacks, and (C,γ) a T-coalgebra. We have an extension $\hat{T}$ of T to a relator. Let an L-algebra $(\mathcal{P}C,[d,e])$ given by

$$\begin{array}{rccc} d: & \mathcal{P}_\alpha\mathcal{P}C & \to & \mathcal{P}C \\ & \mathfrak{C} & \mapsto & \bigcap_{\mathfrak{c}\in\mathfrak{C}}\mathfrak{c} \\ e: & T\mathcal{P}C & \to & \mathcal{P}C \\ & t & \mapsto & \{c\in C \mid (\,\gamma(c),t\,)\in\hat{T}{\in_C}\} \end{array}$$

where $\in_C$ $(\subseteq C\times\mathcal{P}C)$ is the membership relation of C, conceived as a $\mathbb{R}\mathrm{el}$-arrow from C to $\mathcal{P}C$.

Then the semantics $\llbracket\ \rrbracket_{(C,\gamma)}$ of $\mathcal{L}_T^\alpha$ in (C,γ) is defined by the unique morphism of L-algebras from $(\mathcal{L}_T^\alpha,[\bigwedge,\nabla])$ to $(\mathcal{P}C,[d,e])$, which maps a formula ϕ to $\llbracket\phi\rrbracket_{(C,\gamma)}$.

$$\begin{tikzcd}
\mathcal{P}_\alpha\mathcal{L}_T^\alpha \arrow[rrrr, dashed, "\mathcal{P}_\alpha\llbracket\ \rrbracket_{(C,\gamma)}"'] \arrow[dr] \arrow[ddd, bend right, "\bigwedge"'] & & & & \mathcal{P}_\alpha\mathcal{P}C \arrow[dl] \arrow[ddd, bend left, "d"] \\
& L\mathcal{L}_T^\alpha \arrow[rr, dashed, "L\llbracket\ \rrbracket_{(C,\gamma)}"'] \arrow[dd, "{[\bigwedge,\nabla]}"'] & & L\mathcal{P}C \arrow[dd, "{[d,e]}"] & \\
& T\mathcal{L}_T^\alpha \arrow[u] \arrow[d, "\nabla"] \arrow[rr, dashed, "T\llbracket\ \rrbracket_{(C,\gamma)}"'] & & T\mathcal{P}C \arrow[u] \arrow[d, "e"'] & \\
& \mathcal{L}_T^\alpha \arrow[rr, dashed, "\llbracket\ \rrbracket_{(C,\gamma)}"'] & & \mathcal{P}C &
\end{tikzcd}$$

∎

Proposition 4.6 *Suppose that the semantics $\llbracket\ \rrbracket_{(C,\gamma)}$ in (C,γ) is given for the language $\mathcal{L}_T^\alpha$. Then*

$$\begin{array}{lcl} c\models_{(C,\gamma)}\bigwedge\Phi & \textit{iff} & c\models_{(C,\gamma)}\phi \quad \textit{for all } \phi\in\Phi, \\ c\models_{(C,\gamma)}\nabla t & \textit{iff} & (\gamma(c),t)\in\hat{T}{\models_{(C,\gamma)}}, \end{array}$$

where $\hat{T}{\models_{(C,\gamma)}}$ is a relator $\hat{T}$ applied to a $\mathbb{R}\mathrm{el}$-arrow $\models_{(C,\gamma)}$ from C to $\mathcal{L}_T^\alpha$.

Proof. The first clause immediately follows from the commutativity $\llbracket\ \rrbracket_{(C,\gamma)}\circ\bigwedge = d\circ\mathcal{P}_\alpha\llbracket\ \rrbracket_{(C,\gamma)}$.

For the second, the key observation is that $\models_{(C,\gamma)} = (\mathcal{G}_{\llbracket\ \rrbracket_{(C,\gamma)}})^{\mathrm{op}}\circ\in_C$ as a $\mathbb{R}\mathrm{el}$-arrow, which yields that $\hat{T}(\models_{(C,\gamma)}) = (\mathcal{G}_{T\llbracket\ \rrbracket_{(C,\gamma)}})^{\mathrm{op}}\circ\hat{T}{\in_C}$ by the functoriality of $\hat{T}$. Now suppose $c\models_{(C,\gamma)}\nabla t$, which is equivalent to $c\in e\circ(T\llbracket\ \rrbracket_{(C,\gamma)})(t)$. This is rewritten as $(\,\gamma(c),(T\llbracket\ \rrbracket_{(C,\gamma)})(t)\,)\in\hat{T}{\in_C}$ by the definition of e, hence the observation establishes the claim. ∎

Example 4.7 (Stream automata) Consider stream automata over an alphabet L which can be thought of as $(T=L\times\mathrm{id})$-coalgebras. Then T is ω-accessible which preserves weak pullbacks. Here we are concerned with the language $\mathcal{L}_T^\omega$ and its semantics.

Relaxing the definition, we obtain the BNF definition of a formula in $\mathcal{L}_T^\omega$:

$$\mathcal{L}_T^\omega\ \ni\ \phi,\psi\ ::=\phi\wedge\psi\mid\nabla(l,\phi),$$

where $l\in L$.

Given a stream automaton $(C,\langle\mathsf{hd},\mathsf{tl}\rangle)$, $c\models_{(C,\langle\mathsf{hd},\mathsf{tl}\rangle)}\nabla(l,\phi)$ iff $(\,(\mathsf{hd}(c),\mathsf{tl}(c)),(l,\phi)\,)\in\hat{T}(\models_{(C,\langle\mathsf{hd},\mathsf{tl}\rangle)}$ $)$ by the proposition above. It is equivalent to say that there exists some $(l',(c',\phi'))\in T(\models_{(C,\langle\mathsf{hd},\mathsf{tl}\rangle)}) = L\times\models_{(C,\langle\mathsf{hd},\mathsf{tl}\rangle)}$ such that $(T\mathrm{pr}_1)(l',(c',\phi')) = (\mathsf{hd}(c),\mathsf{tl}(c))$ and $(T\mathrm{pr}_2)(l',(c',\phi')) = (l,\phi)$. Since $T=L\times\mathrm{id}$, this means that $l'\in L$, $c'\models_{(C,\langle\mathsf{hd},\mathsf{tl}\rangle)}\phi'$, $(\mathrm{id}_L\times\mathrm{pr}_1)(l',(c',\phi')) = (\mathsf{hd}(c),\mathsf{tl}(c))$ and $(\mathrm{id}_L\times\mathrm{pr}_2)(l',(c',\phi')) = (l,\phi)$. Hence we have established that: $c\models_{(C,\langle\mathsf{hd},\mathsf{tl}\rangle)}\nabla(l,\phi)$ iff $\mathsf{hd}(c)=l$ and $\mathsf{tl}(c)\models_{(C,\langle\mathsf{hd},\mathsf{tl}\rangle)}\phi$. ∎

Adequacy of the language $\mathcal{L}_T^\alpha$ is easy using Lemma 3.2.

Theorem 4.8 (Adequacy) *The language $\mathcal{L}_T^\alpha$ is adequate with respect to T-coalgebras.*

Proof. It suffices to show that $[\![\,]\!]_{(C,\gamma)} = f^{-1} \circ [\![\,]\!]_{(D,\delta)}$ for a morphism $f : (C,\gamma) \to (D,\delta)$. By the definition of $[\![\,]\!]$ as a universal arrow from the initial algebra, we need only to show that $f^{-1} : \mathcal{P}D \to \mathcal{P}C$ is a morphism of $(L = \mathcal{P}_\alpha + T)$-algebras from $(\mathcal{P}D, [d', e'])$ to $(\mathcal{P}C, [d, e])$ which are induced by T-coalgebras (D,δ) and (C,γ), respectively. Hence we are to show that $e \circ T(f^{-1}) = f^{-1} \circ e'$ and $d \circ T(f^{-1}) = f^{-1} \circ d'$. Both are shown by easy calculation; for the first the key is that $\mathcal{G}_{f^{-1}} \circ \in_D \circ \mathcal{G}_f \subseteq \in_C$, $(\mathcal{G}_{f^{-1}})^{\mathrm{op}} \circ \in_C \circ (\mathcal{G}_f)^{\mathrm{op}} \subseteq \in_D$, and $\hat{T}$ is order-preserving (Proposition 4.4). ∎

$\mathcal{L}_T^\alpha$ becomes expressive if α is no less than the accessibility of T;

Theorem 4.9 (Expressivity) *Let T be a κ-accessible $\mathbb{S}\mathrm{et}$-endofunctor which preserves weak pullbacks. Then the language $\mathcal{L}_T^\kappa$ is expressive.*

Proof. It suffices to give a formula ϕ_z^α for each ordinal $\alpha < \kappa$ and each $z \in T^\alpha 1$ such that $[\![\phi_z^\alpha]\!]_{(Z,\zeta)} = (\zeta_\alpha^\kappa)^{-1}(\{z\})$ where (Z,ζ) is the final T-coalgebra and $\zeta_\alpha^\kappa : Z \to T^\alpha 1$ (Lemma 3.3). The definition is by transfinite induction. In the following ζ_α^κ is also designated by ζ_α.

For a limit ordinal α, let $\phi_z^\alpha := \bigwedge_{\beta<\alpha} \phi^\beta_{\zeta_\beta^\alpha(z)}$. It is easy seen that this formula satisfies the condition.

For a successor ordinal $\alpha = \beta + 1$, by the induction hypothesis, we have a map $f : T^\beta 1 \to \mathcal{L}_T^\kappa$ such that for each $w \in T^\beta 1$, $[\![f(w)]\!]_{(Z,\zeta)} = {\zeta_\beta}^{-1}(\{w\})$, i.e. $[\![\,]\!]_{(Z,\zeta)} \circ f = (\zeta_\beta)^{-1} \circ \{\cdot\}_\beta$ where $\{\cdot\}_\beta : T^\beta 1 \to \mathcal{P}T^\beta 1$ maps w to the singleton $\{w\}$. We define $\phi_z^{\beta+1} := \nabla(Tf(z))$. Then, (here $\in_Z$ is the membership relation, a $\mathbb{R}\mathrm{el}$-arrow from Z to $\mathcal{P}Z$)

$$\begin{array}{llll}
u \in [\![\nabla Tf(z)]\!]_{(Z,\zeta)} & \text{iff} & u \in e \circ T[\![\,]\!]_{(Z,\zeta)} \circ Tf(z) & \\
& \text{iff} & (\,\zeta(u),\, T(\zeta_\beta^{-1} \circ \{\cdot\}_\beta)(z)\,) \in \hat{T}\in_Z & [\![\,]\!]_{(Z,\zeta)} \circ f = (\zeta_\beta)^{-1} \circ \{\cdot\}_\beta \\
& \text{iff} & (\zeta(u), z) \in (\mathcal{G}_{T(\zeta_\beta^{-1} \circ \{\cdot\}_\beta)})^{\mathrm{op}} \circ \hat{T}\in_Z & \\
& \text{iff} & (\zeta(u), z) \in \hat{T}(\,(\mathcal{G}_{\zeta_\beta^{-1} \circ \{\cdot\}_\beta})^{\mathrm{op}} \circ \in_Z\,) & \hat{T} \text{ extends } T \\
& \text{iff} & (\zeta(u), z) \in \hat{T}\mathcal{G}_{\zeta_\beta} = \mathcal{G}_{T\zeta_\beta} & (\mathcal{G}_{\zeta_\beta^{-1} \circ \{\cdot\}_\beta})^{\mathrm{op}} \circ \in_Z = \mathcal{G}_{\zeta_\beta} \\
& \text{iff} & z = T\zeta_\beta \circ \zeta(u) & \\
& \text{iff} & u \in (\zeta_{\beta+1})^{-1}(\{z\}), & \zeta_{\beta+1} = T\zeta_\beta \circ \zeta
\end{array}$$

hence we have $[\![\phi_z^{\beta+1}]\!]_{(Z,\zeta)} = (\zeta^{\beta+1})^{-1}(\{z\})$. ∎

For now it seems that no proof systems or general frameworks for giving proof systems have been introduced. We conclude this section by summarizing the above observations in the following table:

Signature functor	accessible and preserves weak pullbacks
Syntax	$\mathcal{L}_T^\alpha \ni \phi ::= \bigwedge \Phi \mid \nabla t$, where $\Phi \subseteq \mathcal{L}_T^\alpha$, $\#\Phi < \alpha$ and $t \in T(\mathcal{L}_T^\alpha)$
Adequacy	holds
Expressivity	holds, if the degree of conjunction is no less than the accessibility of the signature functor

5 Logics designed for specifi c signatures

5.1 Overview

Another kind of approaches are introduced in [11], [17] and [16]; they are focused on rather small classes of signature functors, for which coalgebras do not jump far from Kripke models. Hence many of the results on expressivity or completeness of proof systems are obtained by transferring the results in the well-studied area of standard modal logics and Kripke semantics. Here modal languages and proof systems are designed through a syntactical investigation of a signature functor.

[11] considers signature functors of the form

$$T = (E_1 + O_1 \times \mathrm{id})^{I_1} \times \cdots \times (E_n + O_n \times \mathrm{id})^{I_n}.$$

A functor of this form can be thought of as representing the number and the types of methods of a class in object-oriented languages. Then a coalgebra for the functor is a *class* equipped with this type of methods, and a state of a coalgebra is an *object* of the class.

[17] considers the cases where signatures are *polynomial* functors inductively defined by

$$T_1, T_2 \;::=\; \mathrm{id} \mid C \mid T_1 + T_2 \mid T_1 \times T_2 \mid T_1^E,$$

where E is a set and C is a constant functor mapping any set to a set C. [16], in addition, considers *Kripke polynomial* signature functors defined by

$$T \;::=\; \text{(polynomial functor)} \mid \mathcal{P}T,$$

which now includes the functor $\mathcal{P}\mathrm{id} \times \mathcal{P}A$, the signature of Kripke models over the set of atomic formulas A.

In all of the three papers above modal languages and their semantics are given in a concrete way by investigating signatures syntactically, the results on adequacy and expressivity are established,[10] and complete proof systems are presented. The proofs for the completeness of proof systems are basically based on the well-known method of *canonical model construction*, where *maximal consistent* sets of formulas make up the set of possible worlds. In the latter two papers by Rößiger the method is modified to construct a *canonical coalgebra*, whose state is a maximal consistent set. Moreover in [17], hence for (not Kripke) polynomial functors, it is shown that a canonical coalgebra coincides with a final coalgebra; this observation leads to another characterization (which is functional) of final coalgebras.

5.2 Kurz, 2001 TCS

In the rest of this section we take a glance at a method taken in [11], which is the most simple among the three but suffices to show the essence common in them.

The signatures considered here is of the form

$$T = (E_1 + O_1 \times \mathrm{id})^{I_1} \times \cdots \times (E_n + O_n \times \mathrm{id})^{I_n},$$

where all of E_k, O_k and I_k are all constant functors which can be thought of as the set of error messeges, outputs, and inputs of the k-th *method*, respectively.

Example 5.1 (Buffers) Consider a buffer over the set L which have two methods, `store` and `read`. Let S be the set of its internal states, then the two methods are formalized as functions

$$\mathtt{store} : L \times S \to S, \qquad \mathtt{read} : S \to \{\mathtt{error}\} + L \times S,$$

where, if s is the state which designates that the buffer is empty, then $\mathtt{read}(s) = \mathtt{error}$. Hence a buffer is a coalgebra for the signature functor $T = \mathrm{id}^L \times (\{\mathtt{error}\} + L \times \mathrm{id})$. ∎

[10] Since the languages given are finite, when considering Kripke polynomial functors expressivity holds only for *image-fi nite* coalgebras. In the other cases where the powerset functor $\mathcal{P}$ is not considered, a coalgebra is always image-finite, hence it seems natural that we obtain the expressiveness result in view of the result by Hennessy and Milner.

The language, denoted by $\mathcal{L}_T$, is parametrized only by a signature functor. It is noted that functors considered here are all ω-accessible.

Definition 5.2 (Syntax) Let $T = (E_1 + O_1 \times \mathrm{id})^{I_1} \times \cdots \times (E_n + O_n \times \mathrm{id})^{I_n}$. The language $\mathcal{L}_T$ for T-coalgebras is given by the following inductive BNF definition:

$$\mathcal{L}_T \ \ni\ \phi, \psi \ ::= \bot \mid \phi \to \psi \mid (k, i) = a \mid [k, i]\phi,$$

where $k \in [1, n]$, $i \in I_k$ and $d \in E_k + O_k$. Boolean connectives are defined as usual in terms of $\bot$ and $\to$. $\langle k, i\rangle\phi$ is an abbreviation for $\neg[k, i]\neg\phi$. ∎

$\mathcal{L}_T$ is a finitary multi-modal language whose atomic formula is of the form $(k, i) = a$ and whose modality is given by $[k, i]$ for each $k \in [1, n]$ and $i \in I_k$.

The semantics for $\mathcal{L}_T$ is given in terms of the relation $\models_{(C,\gamma)}$, which can immediately be translated to the definition of $[\![\,]\!]_{(C,\gamma)}$:

Definition 5.3 (Semantics) Given a T-coalgebra (C, γ), the relation $\models_{(C,\gamma)}$ between an element of C and a formula in $\mathcal{L}_T$ is defined inductively by:

$$\begin{array}{lll}
c \not\models_{(C,\gamma)} \bot; & & \\
c \models_{(C,\gamma)} \phi \to \psi & \text{iff} & c \not\models_{(C,\gamma)} \phi \quad \text{or} \quad c \models_{(C,\gamma)} \psi; \\
c \models_{(C,\gamma)} (k, i) = a & \text{iff} & (\mathrm{pr}_k \circ \gamma(c))(i) = a; \\
c \models_{(C,\gamma)} [k, i]\phi & \text{iff} & (\mathrm{pr}_k \circ \gamma(c))(i) \in E_k \quad \text{or} \quad \mathrm{pr}_2(\,(\mathrm{pr}_k \circ \gamma(c))(i)\,) \models_{(C,\gamma)} \phi.
\end{array}$$

∎

$(k, i) = a$ is read as the output of the k-th method with an input i is a, and $[k, i]\phi$ as the updated object (or the next state) produced by the k-th method with an input i, if it exists, satisfies a formula ϕ. Note that if the k-th method with an input i outputs an error message, then $[k, i]\phi$ is true for every formula ϕ; this follows the custom of standard modal logic where $\Box$ is read as *in all the next states ...* It immediately follows from the definition that $c \models_{(C,\gamma)} \langle k, i\rangle\top$ iff $(\mathrm{pr}_k \circ \gamma(c))(i) \notin E_k$, and $c \models_{(C,\gamma)} [k, i]\bot$ iff $(\mathrm{pr}_k \circ \gamma(c))(i) \in E_k$.

Example 5.4 (Buffers) Example 5.1 shows that a buffer over L can be thought of as a T-coalgebra where $T = \mathrm{id}^L \times (\{\mathtt{error}\} + L \times \mathrm{id})$. For them the language is given by atomic propositions $(\mathtt{read}) = a$ where $a \in \{\mathtt{error}\} + L$, and modalities $[store, l]$ and $[read]$ where $l \in L$. ∎

Theorem 5.5 (Adequacy) *The language $\mathcal{L}_T$ is adequate.*

Proof. It suffices to show that validity of formulas is invariant under morphisms (Lemma 3.2); this is easy by the induction on the construction of formulas. ∎

The expressivity result is obtained using the translation of a T-coalgebra into a Kripke model and applying the result [3], [7] as to Kripke models with Hennessy-Milner property.

Theorem 5.6 (Expressivity) *The language $\mathcal{L}_T$ is expressive.*

A complete proof system for $\mathcal{L}_T$ is given considering a T-coalgebra as a Kripke model which satisfies several extra conditions, i.e. by adding some extra axioms to the normal modal logic K. Here we assume an extra condition on the signature functor T.

Definition 5.7 (Proof system) Let $T = (E_1 + O_1 \times \mathrm{id})^{I_1} \times \cdots \times (E_n + O_n \times \mathrm{id})^{I_n}$ be a $\mathbb{S}\mathrm{et}$-endofunctor such that each of $E_1, O_1, \ldots, E_n, O_n$ is a finite set, and $\mathcal{L}_T$ the language for T-coalgebras. A proof system Π_T is given by the following axioms and rules, as a Hilbert-style system:

(Taut)	all Boolean tautologies
(K)	$[k,i](\phi \to \psi) \to ([k,i]\phi \to [k,i]\psi)$
(Err)	$(k,i) = e \to [k,i]\bot$ for all $k \in [1,n]$, $i \in I_k$ and $e \in E_k$
(Suc)	$(k,i) = o \to \langle k,i\rangle\top$ for all $k \in [1,n]$, $i \in I_k$ and $o \in O_k$
(USuc)	$\langle k,i\rangle\phi \to [k,i]\phi$
(UOP)	$(k,i) = a \to \neg(\,(k,i) = a'\,)$ for all $k \in [1,n]$, $i \in I_k$, $a, a' \in E_k + O_k$ and $a \neq a'$
(EOP)	$\bigvee_{a \in E_k + O_k} (k,i) = a$

$$\frac{A \to B \quad A}{B}\ \text{(MP)} \qquad\qquad \frac{\phi}{[k,i]\phi}\ \text{(Nec)}$$

We put $\Pi_T \vdash \phi$ if a formula $\phi \in \mathcal{L}_T$ is deduced by the axioms and rules above. ∎

(Taut), Kripke's axiom (K), modus ponens (MP) and necessitation (Nec) together form the normal (multi-)modal logic K. The intension of each of the other axioms is as follows. (Err) is to designate that if the output is an error message, then no updated object (or the successor state) comes about. (Suc) is that, otherwise, there exists an updated object. It is noted that (Suc) is equivalent to the axiom

$$(k,i) = o \to ([k,i]\phi \to \langle k,i\rangle\phi)$$

since $\Diamond\top \leftrightarrow (\Box\phi \to \Diamond\phi)$ in the normal modal logic K. (USuc) is that the successor state is unique, if it exists. (UOP) is that the value of the output, whether it is an error message or not, is unique. (EOP) is for the existence of the output. To make the proof system finitary, we must have $\#(E_k + O_k) < \infty$.

Soundness of the system Π_T is easy by induction.

Theorem 5.8 (Soundness) *Π_T is sound with respect to T-coalgebras.*

Completeness is shown by translating the result on standard modal logic and Kripke models, which is obtained by the *canonical model* construction with maximal consistent sets its possible worlds.

Theorem 5.9 (Completeness) *Π_T is complete with respect to T-coalgebras.*

6 Modalities induced by predicate liftings

6.1 Overview

The other approach, which is first introduced in [8] and developed in e.g. [14] or [13], defines modal languages by means of *predicate liftings*, a notion which appears in the context of *fibrations*.[11] The method presented in this section considers a large class of signature functors, including $\mathcal{P}$ which Moss's approach cannot handle, and the syntax follows the custom of standard modal languages. The modal languages and proof systems introduced here are parametrized by many parameters, and the results on expressivity or completeness are given in the form that *if the language/system is given in the good manner, i.e. the parameters satisfy certain conditions, then expressivity/completeness holds.*

The section follows the line presented in [15].

[11] What we are interested in can be stated in terms of the fibration $\mathbb{S}\mathrm{ub}\mathbb{S}\mathrm{et} \to \mathbb{S}\mathrm{et}$; the definition of predicate liftings for this specific fibration can be put differently and more easily, which we adopt in the following. The use of predicate liftings in the theory of coalgebra is first presented in [6].

6.2 Predicate liftings

Definition 6.1 (Predicate lifting) Let T be an endofunctor on $\mathbb{S}\mathrm{et}$. A *predicate lifting* λ for T is a family of $\mathbb{S}\mathrm{et}$-arrows $(\lambda_X : \mathcal{P}X \to \mathcal{P}TX)_{X \in \mathbb{S}\mathrm{et}}$ which:

- is order-preserving, i.e. $\mathfrak{x} \subseteq \mathfrak{x}' \subseteq X$ implies $(\lambda_X)(\mathfrak{x}) \subseteq (\lambda_X)(\mathfrak{x}') \subseteq TX$;
- is compatible with inverse images, i.e. for an arbitrary $\mathbb{S}\mathrm{et}$-arrow $f : X \to Y$, $(Tf)^{-1} \circ \lambda_Y = \lambda_X \circ f^{-1}$.[12]

$$\begin{tikzcd}
\mathcal{P}Y \arrow[r, "\lambda_Y"] \arrow[d, "f^{-1}"'] & \mathcal{P}TY \arrow[d, "(Tf)^{-1}"] \\
\mathcal{P}X \arrow[r, "\lambda_X"'] & \mathcal{P}TX
\end{tikzcd}$$

∎

In other words, λ is an order-preserving natural transformation from $\mathcal{P}$ to $\mathcal{P}T$, where $\mathcal{P}$ is the *contravariant* powerset functor.

Informally, a predicate lifting λ translates a predicate in X (i.e. a subset of X) into a predicate of TX, *with respect to a certain aspect*. The following examples may be illustrative:

Example 6.2 (Predicate liftings for $T = L \times \mathrm{id}$) Note that a T-coalgebra is a stream automaton over the alphabet L.

We can define a predicate lifting λ for T by

$$\begin{array}{rccc}
\lambda_X : & \mathcal{P}X & \to & \mathcal{P}TX \\
& \mathfrak{x} & \mapsto & \{\,(l', x') \in TX \mid x' \in \mathfrak{x}\}
\end{array}$$

It is easily verified that λ is indeed a predicate lifting. So to speak, λ extracts the aspect of location.

Another predicate lifting λ_l can be defined for each $l \in L$, by

$$\begin{array}{rccc}
(\lambda_l)_X : & \mathcal{P}X & \to & \mathcal{P}TX \\
& \mathfrak{x} & \mapsto & \{\,(l', x') \mid l' = l\}
\end{array}$$

λ_l extracts which letter is displayed. ∎

Example 6.3 (Predicate liftings for $T = \mathcal{P}\mathrm{id} \times \mathcal{P}A$) A T-coalgebra is a Kripke model over the set of atomic formulas A.

We can define a predicate lifting λ for T by

$$\begin{array}{rccc}
\lambda_X : & \mathcal{P}X & \to & \mathcal{P}TX \\
& \mathfrak{x} & \mapsto & \{(\mathfrak{x}', \mathfrak{a}) \in TX \mid \mathfrak{x}' \subseteq \mathfrak{x}\}
\end{array}$$

λ extracts the aspect of location.

Another predicate lifting λ_a can be defined for each $a \in A$, by

$$\begin{array}{rccc}
(\lambda_a)_X : & \mathcal{P}X & \to & \mathcal{P}TX \\
& \mathfrak{x} & \mapsto & \{(\mathfrak{x}', \mathfrak{a}) \in TX \mid a \in \mathfrak{a}\}
\end{array}$$

λ_a extracts the aspect of satisfaction of a. Note that $(\lambda_a)_X$ is a constant map. ∎

[12] This amounts to say that λ is a natural transformation $\mathcal{P} \Rightarrow \mathcal{P}T$, where $\mathcal{P}$ is considered as a contravariant functor which maps $f : A \to B$ to $\mathcal{P}f := f^{-1} : \mathcal{P}B \to \mathcal{P}A$.

The above two examples motivate two principles for obtaining predicate liftings, which is due to Pattinson:

Proposition 6.4 *Let $\mu : T \Rightarrow \mathcal{P}$ be a natural transformation, where $\mathcal{P}$ is covariant. We obtain a predicate lifting λ for T by*

$$\lambda_X(\mathfrak{x}) := \{t \in TX \mid (\mu X)(t) \subseteq \mathfrak{x}\}.$$

Again λ defined in this way is focused on the location.

Proposition 6.5 *Let $\mathfrak{a} \subseteq T1$. We obtain a predicate lifting $\lambda_{\mathfrak{a}}$ by*

$$(\lambda_{\mathfrak{a}})_X(\mathfrak{x}) := \{t \in TX \mid (T!_X)(t) \in \mathfrak{a}\}.$$

Again $\lambda_{\mathfrak{a}} X$ is a constant map. $T1$ can be considered as the set of behaviors observable in a one-step transition, hence the map $T!_X : TX \to T1$ extracts the aspects which are independent from X.

In the following we will introduce a modality induced by a predicate lifting λ; this correspond to $\Box$. And we will put $\langle\lambda\rangle$ for $\neg[\lambda]\neg$ just as we put $\Diamond$ for $\neg\Box\neg$. The modality $\langle\lambda\rangle$ actually has the corresponding predicate lifting, too.

Proposition 6.6 *Let λ be a predicate lifting for T, and $\neg\lambda\neg$ be defined by*

$$(\neg\lambda\neg)_X(\mathfrak{x}) := TX \setminus (\lambda X)(X \setminus \mathfrak{x}).$$

Then $\neg\lambda\neg$ is again a predicate lifting for T.

Proof. Use the fact that inverse images preserves negations. ∎

6.3 Syntax, semantics, adequacy and expressivity

A functor T, a class Λ of predicate liftings for T and a regular cardinal κ give rise to a multi-modal language $\mathcal{L}^\kappa(\Lambda)$ for T-coalgebras.

Definition 6.7 (Syntax) A language $\mathcal{L}^\kappa(\Lambda)$ associated with Λ is defined by

$$\mathcal{L}^\kappa(\Lambda) \ \ni \phi \ ::= \ \bigwedge \Phi \mid \neg\phi \mid [\lambda]\phi,$$

where $\Phi \subseteq \mathcal{L}^\kappa(\Lambda)$, $\#\Phi < \kappa$ and $\lambda \in \Lambda$. ∎

In short, $\mathcal{L}^\kappa(\Lambda)$ is an infinitary multi-modal language which admits conjunction of less than κ formulas and each modality induced by $\lambda \in \Lambda$. Note that $\top := \bigwedge \emptyset$ is a formula.

To interpret a modality $[\lambda]$ we make use of the predicate lifting λ.

Definition 6.8 (Semantics) Let (C, γ) be a T-coalgebra and $\phi \in \mathcal{L}^\kappa(\Lambda)$. The semantics $[\![\phi]\!]_{(C,\gamma)}$ is defined inductively by the following clauses:

$$\begin{aligned}
[\![\textstyle\bigwedge \Phi]\!] &:= \bigcap_{\phi \in \Phi} [\![\phi]\!]; \\
[\![\neg\phi]\!] &:= C \setminus [\![\phi]\!]; \\
[\![\,[\lambda]\phi\,]\!] &:= \gamma^{-1} \circ \lambda_C([\![\phi]\!]).
\end{aligned}$$

∎

Intuitively, the formula $[\lambda]\phi$ is read as *after one step of transition ϕ holds with respect to λ*. This idea is reflected in the definition above; being ϕ (denoted by $[\![\phi]\!]$) is translated into TC by λC (i.e. $\lambda C([\![\phi]\!])$), and then pulled back along the transition map γ.

Example 6.9 (Modal language for stream automata) Consider the language $\mathcal{L}^{\omega}(\{\lambda\} \cup \{\lambda_l \mid l \in L\})$ where λ and λ_l are as defined in Example 6.2. This is a modal language for $(L \times \mathrm{id})$-coalgebras. Relaxing the definition, we have

$$\begin{array}{lcl} s \models_{(S,\langle \mathsf{hd},\mathsf{tl}\rangle)} [\lambda]\phi & \text{iff} & \mathsf{tl}(s) \models_{(S,\langle \mathsf{hd},\mathsf{tl}\rangle)} \phi, \\ s \models_{(S,\langle \mathsf{hd},\mathsf{tl}\rangle)} [\lambda_l]\phi & \text{iff} & \mathsf{hd}(s) = l. \end{array}$$

∎

Example 6.10 (Modal language for Kripke models) Consider $\mathcal{L}^{\omega}(\{\lambda\} \cup \{\lambda_a \mid a \in A\})$ where λ and λ_a are as defined in Example 6.3. This is a modal language for $(\mathcal{P}\mathrm{id} \times \mathcal{P}A)$-coalgebras. Then we have

$$\begin{array}{lcl} s \models_{(S,\langle \mathsf{next},\mathsf{prop}\rangle)} [\lambda]\phi & \text{iff} & s' \models_{(S,\langle \mathsf{next},\mathsf{prop}\rangle)} \phi \quad \text{for all } s' \in \mathsf{next}(s), \\ s \models_{(S,\langle \mathsf{next},\mathsf{prop}\rangle)} [\lambda_a]\phi & \text{iff} & a \in \mathsf{prop}(s). \end{array}$$

Hence $[\lambda]$ correspond to $\Box$, and $[\lambda_a]\phi$ to the atomic formula a. ∎

Adequacy is easily shown using Lemma 3.2.

Theorem 6.11 (Adequacy) *$\mathcal{L}^{\kappa}(\Lambda)$ is adequate.*

Proof. It suffices to show that $[\![\phi]\!]_{(C,\gamma)} = f^{-1} \circ [\![\phi]\!]_{(D,\delta)}$ for a morphism $f : (C,\gamma) \to (D,\delta)$ and $\phi \in \mathcal{L}^{\kappa}(\Lambda)$, which is by induction on the construction of ϕ. ∎

For expressivity, we must recall that the language is parametrized by the degree of conjunction κ and the set of predicate liftings Λ. The larger κ is, or the richer Λ is, the more descriptive $\mathcal{L}^{\kappa}(\Lambda)$ is. Hence we are to consider under what condition on κ and Λ the language $\mathcal{L}^{\kappa}(\Lambda)$ becomes expressive. [14] gives one answer, namely *separating property*.

Definition 6.12 (Separating system of subsets) Let S a set, $\mathfrak{S} \subseteq \mathcal{P}S$, and define a map $b : S \to \mathcal{P}\mathfrak{S}$ by

$$b(s) := \{\mathfrak{s} \in \mathfrak{S} \mid s \in \mathfrak{s}\}.$$

$\mathfrak{S}$ is said to be *separating* if b is an injection. ∎

In the above, $\mathfrak{S}$ is a system of subsets of S and $b(s)$ is the collection of sets in $\mathfrak{S}$ to which s belongs. $\mathfrak{S}$ is separating iff we can separate elements in S (i.e. distinguish an element in S) in terms of $\mathfrak{S}$.

Definition 6.13 (Separating system of predicate liftings) A class Λ of predicate liftings for T is said to be *separating* if for any set X, the system

$$\{\lambda_X(\mathfrak{x}) \mid \lambda \in \Lambda, \mathfrak{x} \in \mathcal{P}X\}$$

is a separating system of subsets of TX. ∎

Example 6.14 (Kripke models) Let $T = \mathcal{P}\mathrm{id} \times \mathcal{P}A$, λ and λ_a ($a \in A$) defined by Example 6.3; that is, λ is for the next state and λ_a is for the satisfaction of a. Then it is easily verified that $\Lambda := \{\lambda\} \cup \{\lambda_a \mid a \in A\}$ is separating.[13] ∎

Now we are to establish that if T is accessible and Λ is separating, then $\mathcal{L}^\sigma(\Lambda)$ is expressive for a sufficiently large cardinal σ. This can be understood intuitively as follows. Let T be κ-accessible, which means that we have only to see behaviors of T-coalgebras within less than κ-steps. The separating Λ gifts us with a set of modalities which is rich enough to express *one-step* transition. Taking conjunction of formulas each of which expresses the behavior within 1-step, 2-steps, ... , we obtain a formula which specifies a behavior within κ-steps.

In view of the sketch of proof stated above, it seems necessary that we can specify one element of TX by taking intersection (which corresponds to conjunction) of the system of subsets induced by Λ. In the general case of separating systems of subsets this is not true; consider $S = \{x, y\}$ and $\mathfrak{S} = \{\{x, y\}, \{y\}\}$, where we cannot take $\{x\}$ as an intersection of sets in $\mathfrak{S}$. However, in this case of predicate liftings, this specification by intersection can be done, with the help of predicate liftings $\neg\lambda\neg$.

Lemma 6.15 *Let X be a set, Λ a separating set of predicate liftings for T,*

$$\begin{aligned}
\bar{\Lambda} &:= \Lambda \cup \{\neg\lambda\neg \mid \lambda \in \Lambda\}, \\
\mathfrak{S} &:= \{\lambda_X(\mathfrak{x}) \mid \lambda \in \bar{\Lambda}, \mathfrak{x} \in \mathcal{P}X\}, \\
b :\quad & \begin{array}{ccc} TX & \to & \mathcal{P}\mathfrak{S} \\ t & \mapsto & \{\mathfrak{s} \in \mathfrak{S} \mid t \in \mathfrak{s}\} \end{array}
\end{aligned}$$

Then for all $t \in TX$, $\{t\} = \bigcap b(t)$.

Proof. It is obvious that $\bar{\Lambda}(\supseteq \Lambda)$ is also separating. We argue by contradiction.

Assume $t' \in \bigcap b(t)$ and $t' \neq t$. $t' \in \bigcap b(t)$ immediately yields that $b(t) \subseteq b(t')$, and since b is monic, $b(t) \subsetneq b(t')$. Take $\lambda \in \bar{\Lambda}$ and $\mathfrak{x} \in \mathcal{P}X$ such that $\lambda_X(\mathfrak{x}) \in b(t') \setminus b(t)$, i.e. $t' \in \lambda_X(\mathfrak{x})$ and $t \notin \lambda_X(\mathfrak{x})$ (†).

$t' \in \lambda_X(\mathfrak{x})$ can be rewritten as $t' \notin (\neg\lambda\neg)_X(X \setminus \mathfrak{x})$, i.e. $(\neg\lambda\neg)_X(X \setminus \mathfrak{x}) \notin b(t')$, which shows $(\neg\lambda\neg)_X(X \setminus \mathfrak{x}) \notin b(t)$ by $b(t) \subsetneq b(t')$. Hence we obtain $t \in \lambda_X(\mathfrak{x})$ which contradicts (†). ∎

However, the lemma is not enough in that the degree of intersection (i.e. conjunction, on the logical side) is not bounded above. Fortunately, if T is κ-accessible we can do better:

Lemma 6.16 *Let T be κ-accessible, Λ separating, X a set and $t \in TX$. Then there exists $\mathfrak{x}_t(\subseteq X)$ such that $\#\mathfrak{x}_t < \kappa$ and*

$$\{t\} = \bigcap\{\lambda_X(\mathfrak{x}) \mid \lambda \in \bar{\Lambda}, \mathfrak{x} \subseteq \mathfrak{x}_t, t \in \lambda_X(\mathfrak{x})\}.$$

Proof. Take $\mathfrak{x}_t$ (with the inclusion map $i : \mathfrak{x}_t \rightarrowtail X$) as $t \in (Ti)[T\mathfrak{x}_t]$, which is possible by the accessibility of T. We are to show that this $\mathfrak{x}_t$ satisfies the condition.

Lemma 6.15 above shows that $\{t\} = \bigcap\{\lambda_X(\mathfrak{x}) \mid \lambda \in \bar{\Lambda}, \mathfrak{x} \subseteq X, t \in \lambda_X(\mathfrak{x})\}$, and note that the statement can be put as $\{t\} = \bigcap\{\lambda_X(\mathfrak{x}\cap\mathfrak{x}_t) \mid \lambda \in \bar{\Lambda}, \mathfrak{x} \subseteq X, t \in \lambda_X(\mathfrak{x}\cap\mathfrak{x}_t)\}$. The order-preserving property of predicate liftings yields $\lambda_X(\mathfrak{x} \cap \mathfrak{x}_t) \subseteq \lambda_X(\mathfrak{x})$, hence it suffices to show that for every $\mathfrak{x} \subseteq X$, if $t \in \lambda_X(\mathfrak{x})$ then $t \in \lambda_X(\mathfrak{x} \cap \mathfrak{x}_t)$.

Let $s \in T\mathfrak{x}_t$ be such that $t = (Ti)(s)$. Then $(Ti)(s) \in \lambda_X(\mathfrak{x})$ i.e. $s \in (Ti)^{-1} \circ \lambda_X(\mathfrak{x})$, hence by the compatibility of predicate liftings $s \in \lambda_{\mathfrak{x}_t} \circ i^{-1}(\mathfrak{x})$. Using $i^{-1}(\mathfrak{x}) = i^{-1}(\mathfrak{x} \cap \mathfrak{x}_t)$ and going up along the same argument, we have $t \in \lambda_X(\mathfrak{x} \cap \mathfrak{x}_t)$. ∎

Theorem 6.17 (Expressivity) *Let T be κ-accessible, Λ separating and σ a regular cardinal which is: $\sigma > \#\Lambda$, and $\sigma > 2^\alpha$ for all $\alpha < \kappa$. Then the modal language $\mathcal{L}^\sigma(\Lambda)$ is expressive.*

[13]Nevertheless, the separating property of Λ does not imply expressivity since T is not accessible.

Proof. We construct a formula ϕ_z^α for each $\alpha < \kappa$ and $z \in T^\alpha 1$ in Lemma 3.3, and then use the lemma.

For a limit ordinal α, $z \in T^\alpha 1$ can be written as $z = (z_\beta)_{\beta<\alpha}$ by the set-theoretical characterization of the limit $T^\alpha 1$. Take ϕ_z^α as

$$\phi_z^\alpha := \bigwedge_{\beta<\alpha} \phi_{z_\beta}^\beta.$$

Then by the induction hypothesis $[\![\phi_z^\alpha]\!]_{(Z,\zeta)} = \bigcap_{\beta<\alpha} \zeta_\beta^{-1}(z_\beta)$, which is equal to $\zeta_\alpha^{-1}(\{z\})$.

For a successor ordinal $\alpha = \beta + 1$, apply Lemma 6.16 to $z \in T(T^\beta 1)$ and obtain $\mathfrak{b}_z \subseteq T^\beta 1$ such that: $\#\mathfrak{b}_z < \kappa$ and

$$\{z\} = \bigcap\{\lambda_{T^\beta 1}(\mathfrak{b}) \mid \lambda \in \bar{\Lambda}, \mathfrak{b} \subseteq \mathfrak{b}_z, z \in \lambda_{T^\beta 1}(\mathfrak{b})\}.$$

Let $\mathfrak{B}_\lambda := \{\mathfrak{b} \subseteq \mathfrak{b}_z \mid z \in \lambda_{T^\beta 1}(\mathfrak{b})\}$ and $\mathfrak{B}'_\lambda := \{\mathfrak{b} \subseteq \mathfrak{b}_z \mid z \in (\neg\lambda\neg)_{T^\beta 1}(\mathfrak{b})\}$ for each $\lambda \in \Lambda$. Then the equation above can be put as

$$\{z\} = \left[\bigcap_{\lambda\in\Lambda} \bigcap_{\mathfrak{b}\in\mathfrak{B}_\lambda} \lambda_{T^\beta 1}(\mathfrak{b})\right] \cap \left[\bigcap_{\lambda\in\Lambda} \bigcap_{\mathfrak{b}\in\mathfrak{B}'_\lambda} (\neg\lambda\neg)_{T^\beta 1}(\mathfrak{b})\right].$$

Now let

$$\phi_z^\alpha := \left[\bigwedge_{\lambda\in\Lambda} \bigwedge_{\mathfrak{b}\in\mathfrak{B}_\lambda} [\lambda]\left(\bigvee_{b\in\mathfrak{b}} \phi_b^\beta\right)\right] \wedge \left[\bigwedge_{\lambda\in\Lambda} \bigwedge_{\mathfrak{b}\in\mathfrak{B}'_\lambda} \langle\lambda\rangle\left(\bigvee_{b\in\mathfrak{b}} \phi_b^\beta\right)\right].$$

This is admitted to be a formula in $\mathcal{L}^\sigma(\Lambda)$ since σ is larger than $\#\Lambda$, $\#\mathfrak{B}_\lambda$ and $\#\mathfrak{b}$, and it is easy to see that $[\![\phi_z^\alpha]\!]_{(Z,\zeta)} = \zeta_\alpha^{-1}(\{z\})$. ∎

The degree of intersection σ in the theorem seems much larger than κ; however, even finitary modal languages are expressive in some settings:

Corollary 6.18 *If T is ω-accessible and Λ is a separating set of predicate liftings with $\#\Lambda < \omega$, then $\mathcal{L}^\omega(\Lambda)$ is expressive.*

If T is well-behaved in the sense defined below, we can reduce σ to κ.

Definition 6.19 (Intersection preserving predicate lifting) A predicate lifting λ for T is *intersection preserving* if for any set X and any $\mathfrak{X} \subseteq \mathcal{P}X$,

$$\lambda_X(\bigcap\mathfrak{X}) = \bigcap_{\mathfrak{x}\in\mathfrak{X}} \lambda_X(\mathfrak{x}).$$

Λ is *intersection preserving* if so is every $\lambda \in \Lambda$. ∎

Proposition 6.20 *Predicate liftings defined by the principles in Proposition 6.4, 6.5 are intersection preserving.*

Proof. Easy. ∎

Theorem 6.21 *Let T be κ-accessible, Λ separating, intersection preserving and $\#\Lambda < \kappa$. Then the modal language $\mathcal{L}^\kappa(\Lambda)$ is expressive.*

Proof. We modify the proof of Theorem 6.17, reducing the degree of conjunction or disjunction. The problem lies in defining ϕ_z^α for a successor ordinal α.

By the intersection preserving property of $\lambda \in \Lambda$, we have

$$\bigcap_{\lambda\in\Lambda} \bigcap_{\mathfrak{b}\in\mathfrak{B}_\lambda} \lambda_{T^\beta 1}(\mathfrak{b}) = \bigcap_{\lambda\in\Lambda} \lambda_{T^\beta 1}(\bigcap\mathfrak{B}_\lambda),$$

and since, as easily seen, $\neg\lambda\neg$ preserves unions for intersection preserving λ, we have

$$\begin{aligned}\bigcap_{\lambda\in\Lambda}\bigcap_{\mathfrak{b}\in\mathfrak{B}'_\lambda}(\neg\lambda\neg)_{T^\beta 1}(\mathfrak{b}) &= \bigcap_{\lambda\in\Lambda}\bigcap_{\mathfrak{b}\in\mathfrak{B}'_\lambda}\bigcup_{b\in\mathfrak{b}}(\neg\lambda\neg)_{T^\beta 1}(\{b\})\\ &= \bigcap_{\lambda\in\Lambda}\bigcap_{\{b\}\in\mathfrak{B}'_\lambda}(\neg\lambda\neg)_{T^\beta 1}(\{b\}).\end{aligned}$$

Hence we can define

$$\phi_z^\alpha := \left[\bigwedge_{\lambda\in\Lambda}[\lambda]\left(\bigvee_{b\in\bigcap\mathfrak{B}_\lambda}\phi_b^\beta\right)\right]\wedge\left[\bigwedge_{\lambda\in\Lambda}\bigwedge_{\{b\}\in\mathfrak{B}'_\lambda}\langle\lambda\rangle\phi_b^\beta\right],$$

which is a formula in $\mathcal{L}^\kappa(\Lambda)$ since $\#\bigcup\mathfrak{B}_\lambda$ and $\#\bigcup\mathfrak{B}'_\lambda$ are smaller than κ. ▮

Example 6.22 (Finitely branching labelled transition systems) Let L be a finite set of labels and $T := \mathcal{P}_\omega(L\times\mathrm{id})$, where $\mathcal{P}_\omega$ is the powerset functor bounded by ω. Then a T-coalgebra is a finitely branching labelled transition system with the set of labels L.

Consider a natural transformation $\mu_l : T \Rightarrow \mathcal{P}$ for each $l \in L$ defined by

$$(\mu_l)_X((l_i, x_i)_{i\in[1,n]}) := \{x_i \mid l_i = l\}.$$

Then the principle presented in Proposition 6.4 induces a predicate lifting λ_l for each $l \in L$. The modality $[\lambda_l]$ is quite natural in that $c \models_{(C,\gamma)} [\lambda_l]\phi$ holds iff $c' \models_{(C,\gamma)} \phi$ holds for all $c' \in C$ such that $c \xrightarrow{l} c'$.

Now it is easy to see that T is ω-accessible, $\Lambda := \{\lambda_l \mid l \in L\}$ separating and $\#\Lambda < \omega$, hence Corollary 6.18 yields the expressivity of $\mathcal{L}^\omega(\Lambda)$; this re-proves the result in [5] in the coalgebraic framework. ▮

6.4 Proof system, soundness and completeness

Now we are concerned with proof systems for the modal logics introduced in the previous subsection. [13] introduces results on soundness and completeness of proof systems which

- are parametrized by a set of axiom schema, and
- handle only one-step transitions, i.e. axiom schema do not have nesting modalities,

under certain conditions on a set of axiom schema.

The conditions under which soundness or completeness holds are rather complicated, so here we present only the sketch of the results and the proofs for them.

We are focused only on finitary modal languages (which, of course, may fail to be expressive).

The key is that both validity of a formula (denoted by $\models \phi$) and deducibility by the system of a formula (denoted by $\vdash \phi$) can be decomposed into n-step versions of each. For example, if the degree of nesting modalities in a formula ϕ is n, then ϕ refers only to behaviors within n-step transitions, hence we can know the validity of ϕ by the n-step validity (denoted by $\models^n \phi$). For deducibility, the decomposition into n-step versions is possible since axiom schema refer only to one-step transitions, and the deducibility of ϕ whose degree of nesting modalities is n is reduced to the n-step deducibility of ϕ (denoted by $\vdash^n \phi$).

Then the results on completeness and soundness are given in the form that *if the set of axiom schema is sound/complete as to one-step transitions, then the proof system is sound/complete*. The proofs are made by showing $\models^n \phi$ iff $\vdash^n \phi$ for every $n \in \omega$, using induction on n.

It may seem that the statement be tautological; *if the system is sound/complete, then it is sound/complete*. However, at least in an example presented by Pattinson it really works. In the example a proof system for the language $\mathcal{L}^\omega(\{\lambda\}\cup\{\lambda_a \mid a \in A\})$ as given in Example 6.10 is introduced, the set of axiom schema is shown to be sound and complete with respect to one-step transitions (which is considerably easy), and then the results are applied to obtain soundness and completeness of the system. This provides another proof (by induction) for Kripke-completeness of normal modal logic $\mathbf{K}$. It is obvious that this method cannot be applied to modal logics such as $\mathbf{S4}$, with nesting modalities in their axiom schema.

Acknowledgement

This report was written during the author's stay at Laboratory for Verification and Semantics, National Institute of Advanced Industrial Science and Technology, Japan (AIST). He is grateful to Hiroshi WATANABE for making the stay possible and a lot of invaluable comments, Dirk PATTINSON for a series of instructive lectures on coalgebras at NASSLLI 2003, Izumi TAKEUTI for advising the author to visit AIST, and last but not least, Ryo KASHIMA for his supervision.

References

[1] Richard Bird and Oege de Moor. *Algebra of Programming*. Prentice Hall, 1997.

[2] A. Carboni, G.M. Kelly, and R.J. Wood. A 2-categorical approach to change of base and geometric morphisms I. Technical Report 90-1, Department of Pure Mathematics, University of Sydney, 1990. ISSN 1033-2359.

[3] Robert Goldblatt. Saturation and the Hennessy-Milner property. In A. Ponse, M. de Rijke, and Y. Venema, editors, *Modal Logic and Process Algebra*, volume 53 of *CSLI Lecture Notes*. Center for the Study of Language and Information, Stanford University, 1995.

[4] Ichiro Hasuo. Modal logics for coalgebras. manuscript, July 2003. `http://www.is.titech.ac.jp/~hasuo2`.

[5] Matthew Hennessy and Robin Milner. On observing nondeterminism and concurrency. In J.W. de Bakker and J. van Leeuwen, editors, *Automata, Languages and Programming, 7th Colloquium*, volume 85 of *Lecture Notes in Computer Science*, pages 299–309. Springer-Verlag, 1980.

[6] Claudio Hermida and Bart Jacobs. Structural induction and coinduction in a fibrational setting. *Information and Computation*, 145:107–152, 1998.

[7] Marco Hollenberg. Hennessy-Milner classes and process algebra. In A. Ponse, M. de Rijke, and Y. Venema, editors, *Modal Logic and Process Algebra*, volume 53 of *CSLI Lecture Notes*. Center for the Study of Language and Information, Stanford University, 1995.

[8] Bart Jacobs. Many-sorted coalgebraic modal logic: a model-theoretic study. *Theoretical Informatics and Applications*, 35(1):31–59, 2001.

[9] Alexander Kurz. *Logics for Coalgebras and Applications to Computer Science*. PhD thesis, Universität München, April 2000.

[10] Alexander Kurz. Coalgebras and modal logic. Lecture notes of ESSLLI 2001, 2001. `http://www.helsinki.fi/esslli/`.

[11] Alexander Kurz. Specifying coalgebras with modal logic. *Theoretical Computer Science*, 260(1-2):119–138, 2001.

[12] Lawrence S. Moss. Coalgebraic logic. *Annals of Pure and Applied Logic*, 96:277–317, 1999.

[13] Dirk Pattinson. Coalgebraic modal logic: Soundness, completeness and decidability of local consequence. *Theoretical Computer Science*, to appear, 2003.

[14] Dirk Pattinson. Expressive logics for coalgebras via terminal sequence induction. *Notre Dame Journal of Formal Logic*, to appear, 2003.

[15] Dirk Pattinson. An introduction to the theory of coalgebras. Lecture notes of NASSLLI 2003, 2003. `http://www.indiana.edu/˜nasslli/`.

[16] Martin Rößiger. Coalgebras and modal logic. In Horst Reichel, editor, *Coalgebraic Methods in Computer Science (CMCS 2000)*, volume 33 of *Electronic Notes in Theoretical Computer Science*, 2000.

[17] Martin Rößiger. From modal logic to terminal coalgebras. *Theoretical Computer Science*, 260:209–228, 2001.

[18] J.J.M.M. Rutten. Relators and metric bisimulations. In B. Jacobs, L. Moss, H. Reichel, and J. Rutten, editors, *Coalgebraic Methods in Computer Science (CMCS '98)*, volume 11 of *Electronic Notes in Theoretical Computer Science*, pages 1–7, 1998.

[19] J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249:3–80, 2000.

[20] Krister Segerberg. An essay in classical modal logic. *Filosofiska Studier*, 13, 1971.

[21] Johan van Benthem. *Modal Correspondence Theory*. PhD thesis, University of Amsterdam, 1976.