Modularity in Coalgebra

Corina Cîrstea [1,2]

University of Southampton, UK

Abstract

This paper gives an overview of recent results concerning the modular derivation of (i) modal specification logics, (ii) notions of simulation together with logical characterisations, and (iii) sound and complete axiomatisations, for systems modelled as coalgebras of functors on Set. Our approach applies directly to an inductively-defined class of coalgebraic types, which subsumes several types of discrete state-based systems, including (probabilistic) transition systems, probabilistic automata and spatial transition systems.

Key words: Coalgebra, modularity, simulation, modal logic, expressiveness, soundness, completeness.

1 Introduction

Following Rutten's seminal paper on universal coalgebra [18], the use of coalgebras as a general, uniform framework for modelling and reasoning about state-based, dynamical systems has become an established field of research. Early work in this area focused on developing the theory of coalgebras at a level of generality that is parametric in the coalgebraic type of interest, with coalgebraic bisimulation, its associated corecursion/coinduction principles, and the study of logics able to express bisimulation-invariant properties of coalgebraic models being central to this work. In contrast, much of the recent and ongoing work is concerned with exploiting/adapting coalgebraic concepts and techniques in order to provide semantic models, logics and reasoning principles for particular classes of systems, including concurrent, probabilistic and mobile systems.

These two different perspectives in the study of coalgebraic models are perhaps most apparent when investigating the relationship between coalgebras and modal logic. On the one hand, the coalgebraic logic of Moss [15] provides a uniform way of defining an expressive, if infinitary logic for coalgebras, while

[1] Partially supported by EPSRC research grant EP/D000033/1.

[2] Email: cc2@ecs.soton.ac.uk

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

imposing essentially no restrictions on the coalgebraic type under consideration. On the other hand, in practical applications, one typically looks for finitary specification logics which are closer in spirit to, say, Hennessy-Milner logic, and which admit complete axiomatisations. Similarly, the coalgebraic notion of bisimulation and the related notion of behavioural equivalence, both defined uniformly on coalgebraic types, turn out to be too restrictive when attempting to capture concrete process equivalences, as employed by existing process calculi. Alternative, more ad hoc notions of simulation are needed to account for the various degrees of observability present in semantic models for such calculi, while the associated reasoning principles and the logics capable of characterising these notions become less canonical.

The approach presented here advocates a modular, systematic approach to the process of deriving (i) coalgebraic notions of process equivalence/refinement which are closer to the needs of concrete specification formalisms, (ii) modal logics which characterise these notions, and (iii) sound and complete axiomatisations for these logics. Our approach involves breaking the uniformity in the underlying functor, and exploiting the structure of the functor in order to derive useful notions of refinement/simulation and suitable specification logics in a modular fashion. Modularity is therefore understood at the meta-level of system types: we first consider a number of simple coalgebraic types, modelling basic aspects of systems such as (non-)deterministic or probabilistic behaviour, with varying degrees of observability, and subsequently provide uniform methodologies for dealing with types obtained by applying a number of type-building operators to these basic coalgebraic types.

Specifically, we show that coalgebraic type-building operators including cartesian products, coproducts and functor composition can naturally be lifted to a relational as well as a logical level, ultimately allowing the derivation of notions of simulation and of corresponding logics for (coalgebras of) functor combinations, from similar notions and logics for the functors being combined. Furthermore, relevant properties of the resulting logics, including expressiveness w.r.t. a particular notion of simulation and the existence of a sound and complete axiomatisation, can themselves be derived in the same modular fashion. The key idea here is to regard the observable behaviour of a system as the successive unfolding of its one-step behaviour, and to restrict attention to this one-step behaviour when lifting the various type-building operators to a relational/logical level, and when formulating conditions that guarantee the existence of logical characterisations and of suitable axiomatisations.

Our results can be used to derive observational equivalences/preorders, together with logical characterisations and complete axiomatisations, for an inductively-defined class of coalgebraic types which is sufficiently general to account for (combinations of) non-deterministic and probabilistic behaviour, as well as for spatial and epistemic aspects of systems. A consequence of our modular approach is that, in order to define a notion of process equivalence together with a logic which characterises it and admits a complete axiomati-

sation, it suffices to treat each of the individual features of the systems being modelled *in isolation*, and specify, for each such feature: (i) what it means for a system to simulate the *one-step* behaviour of another system, and (ii) how such *one-step* behaviours can be logically characterised and axiomatised.

This paper gathers results from previous work [3,4,6,5] concerning the modular derivation of notions of simulation and modal logics, as well as of expressiveness results and of sound and complete axiomatisations, for systems modelled as coalgebras. The paper is structured as follows: Section 2 recalls some basic coalgebraic concepts, illustrating their relevance to the modelling of state-based, dynamical systems. Section 3 reviews existing work in the study of modal logics for coalgebras, and subsequently describes a modular approach to deriving such logics, which subsumes all previously-reviewed approaches. Section 4 summarises existing results on modularly deriving notions of simulation, while Section 5 focuses on modularly deriving logical characterisations for these notions. Next, Section 6 shows how sound and complete axiomatisations for modularly-defined logics can themselves be obtained in a modular fashion. Ongoing and future work are discussed in Section 7.

Acknowledgement

The work described in Section 6 of this paper was carried out jointly with Dirk Pattinson.

2 Preliminaries

In the coalgebraic approach to modelling systems, functors $\mathbb{T} : \mathsf{Set} \to \mathsf{Set}$ are used to structure the information that can be observed about the states of (a certain type of) dynamical systems[3]. A $\mathbb{T}$-*coalgebra*, modelling a particular such system, is given by a pair (C, γ), with C a set (the *carrier* of the coalgebra) and $\gamma : C \to \mathbb{T}C$ a function (the *coalgebra map*). The carrier of the coalgebra models the state space of the system, whereas the coalgebra map gives, for each state, its immediate (one-step) behaviour. A *coalgebra morphism* between $\mathbb{T}$-coalgebras (C, γ) and (D, δ) is given by a map $f : C \to D$ which is structure-preserving, that is, $\mathbb{T}f \circ \gamma = \delta \circ f$. The category of $\mathbb{T}$-coalgebras and coalgebra morphisms is denoted $\mathsf{Coalg}(\mathbb{T})$.

Throughout the paper, $\mathbb{T} : \mathsf{Set} \to \mathsf{Set}$ will denote a (*weak-pullback preserving*[4]) endofunctor on the category of sets. Since $\mathbb{T}$-coalgebras constitute the object of our study, we will often refer to such functors as *coalgebraic types*. The restriction to weak-pullback preserving functors will only be required to derive logical characterisation results for coalgebraic (bi)simulations. However, as shown in [8], weak-pullback preservation is a reasonable assumption

[3] Endofunctors on categories other than Set are also considered in the general theory of coalgebras. However, in this paper we restrict ourselves to endofunctors on Set.

[4] That is, $\mathbb{T}$ transforms pullback diagrams into weak-pullback diagrams.

on coalgebraic types; in its absence, the resulting notion of bisimulation lacks many desirable properties, such as preservation under relational composition.

Coalgebraic bisimulation provides a canonical notion of observational equivalence, which can be defined uniformly on coalgebraic types. Here we give a re-formulation of its original definition [1], which uses the lifting of the functor $\mathbb{T}$ to a category of relations. To this end, we let Rel denote the category having tuples (A, B, R) with $A, B \in |\mathsf{Set}|$ and $R \subseteq A \times B$ as objects, and pairs (f, g) with $f : A \to C$ and $g : B \to D$ being such that $a\, R\, b$ implies $f(a)\, S\, g(d)$, as arrows from (A, B, R) to (C, D, S). For a relation $R \subseteq A \times B$, we write $\pi_1^R : R \to A$ and $\pi_2^R : R \to B$ for the corresponding projection maps. In this setting, relational composition can be defined using a pullback construction in Set. The *lifting* $\Gamma_{\mathbb{T}} : \mathsf{Rel} \to \mathsf{Rel}$ of the functor $\mathbb{T}$ to the category Rel is then defined by $\Gamma_{\mathbb{T}}(A, B, R) = (\mathbb{T}A, \mathbb{T}B, S)$, where $S \subseteq \mathbb{T}A \times \mathbb{T}B$ is the image of $\mathbb{T}R$ under the map $\langle \mathbb{T}\pi_1^R, \mathbb{T}\pi_2^R \rangle : \mathbb{T}R \to \mathbb{T}A \times \mathbb{T}B$. An immediate property of $\Gamma_{\mathbb{T}}$ is the preservation of equality relations and of relational composition.

A *(coalgebraic) bisimulation* between two $\mathbb{T}$-coalgebras (C, γ) and (D, δ) can now be defined simply as a $\Gamma_{\mathbb{T}}$-coalgebra having the pair (γ, δ) as a coalgebra map. Thus, a $\mathbb{T}$-bisimulation between (C, γ) and (D, δ) is given by a map in Rel of the form $\rho : (C, D, R) \to (\mathbb{T}C, \mathbb{T}D, \Gamma_{\mathbb{T}}R)$, or equivalently, by a relation $R \subseteq C \times D$ which is preserved by the coalgebra maps γ and δ. The largest bisimulation between (C, γ) and (D, δ), obtained as the union of all such bisimulations, is denoted by $\simeq$ and is called $\mathbb{T}$-*bisimilarity*.

Of great importance in the study of coalgebraic models are *final coalgebras*, the carriers of which can typically be obtained via a limit construction. For a functor $\mathbb{T} : \mathsf{C} \to \mathsf{C}$ on a complete category C, the *final sequence of* $\mathbb{T}$ is an ordinal-indexed sequence (Z_α) of C-objects, together with a family $(p_\beta^\alpha : Z_\alpha \to Z_\beta)_{\beta \leq \alpha}$ of C-arrows, defined by:

- $Z_0 = 1$, with 1 denoting a final object in C,
- $p_0^\alpha : Z_\alpha \to 1$ is the unique such arrow,
- $Z_{\alpha+1} = \mathbb{T}Z_\alpha$,
- $p_{\beta+1}^{\alpha+1} = \mathbb{T}p_\beta^\alpha$ for $\beta \leq \alpha$,
- $p_\alpha^\alpha = 1_{Z_\alpha}$,
- $p_\gamma^\alpha = p_\gamma^\beta \circ p_\beta^\alpha$ for $\gamma \leq \beta \leq \alpha$,
- if α is a limit ordinal, the cone $Z_\alpha, (p_\beta^\alpha)_{\beta<\alpha}$ for $(p_\gamma^\beta)_{\gamma \leq \beta < \alpha}$ is limiting.

If $\mathbb{T} : \mathsf{C} \to \mathsf{C}$ is an accessible, monic-preserving functor on a locally presentable category, then the final sequence of $\mathbb{T}$ *stabilises at some* α [5], and moreover, Z_α is the carrier of a final $\mathbb{T}$-coalgebra (see [21]). In what follows, this result will be instantiated with endofunctors $\mathbb{T}$ on the categories Set and Rel.

Each $\mathbb{T}$-coalgebra (C, γ) determines a cone $(\gamma_\alpha : C \to Z_\alpha)$ over the final sequence of $\mathbb{T}$. This is defined by letting (i) $\gamma_0 : C \to 1$ to be the unique such

[5] That is, $p_\alpha^{\alpha+1} : Z_{\alpha+1} \to Z_\alpha$ is an isomorphism.

map, (ii) $\gamma_{\alpha+1} = \mathbb{T}\gamma_\alpha \circ \gamma$, and (iii) γ_α to be the unique map satisfying $p^\alpha_\beta \circ \gamma_\alpha = \gamma_\beta$ for each $\beta < \alpha$, if α is a limit ordinal. For an ordinal α, the α-element of the final sequence of $\mathbb{T}$ describes all the possible $\mathbb{T}$-behaviours observable through α unfoldings of the coalgebra map, while the map $\gamma_\alpha : C \to Z_\alpha$ takes states of the coalgebra to their partial behaviour observable in α steps.

For weak-pullback preserving endofunctors $\mathbb{T}$ whose final sequence stabilises, it is possible to give a characterisation of $\mathbb{T}$-bisimulation between two $\mathbb{T}$-coalgebras (C, γ) and (D, δ) in terms of these partial observable behaviours. Specifically, $\simeq$ can be characterised as $\bigcap_\alpha \simeq_\alpha$, where the *$\alpha$-step observability relation* $\simeq_\alpha \subseteq C \times D$ is defined by: $c \simeq_\alpha d$ iff $\gamma_\alpha(c) = \delta_\alpha(d)$, for $c \in C$ and $d \in D$. (Details can be found e.g. in [6].)

By considering an inductively-defined class of endofunctors on Set, one can recover, as $\mathbb{T}$-coalgebras, many interesting types of systems, including (probabilistic) transition systems, (probabilistic) automata, spatial and epistemic models (with or without update). The focus of this paper is on coalgebraic types $\mathbb{T}$ constructed from a small number of basic types (modelling deterministic, non-deterministic and probabilistic behaviour), using a small number of type-building operators. The results in this paper thus apply directly to coalgebras of functors $\mathbb{T}$ generated by the following syntax:

$$\mathbb{T} \ ::= \ C \mid \mathsf{Id} \mid \mathcal{P}_\omega \mid \mathcal{D} \mid \mathcal{S} \mid \mathbb{T} \times \mathbb{T} \mid \mathbb{T} + \mathbb{T} \mid \mathbb{T}^A \mid \mathbb{T} \circ \mathbb{T} \tag{1}$$

where C denotes the constant functor mapping any set to the set C, Id is the identity functor, $\mathcal{P}_\omega$ is the finite powerset functor, $\mathcal{D}$ (resp. $\mathcal{S}$) is the finite (sub-)probability distribution functor, mapping a set to the set of finite (sub-)probability distributions over it, and $\_ \times \_$, $\_ + \_$, $(\_)^A$, and $\_ \circ \_$ denote product, coproduct, exponentiation with fixed exponent A and composition (of functors). However, the general techniques developed here do not rely on the particular shape of these functors, and are easily extendable to more general classes of inductively-defined coalgebraic types.

Example 2.1 (i) Deterministic systems can be modelled as coalgebras of the functor Id^A.

(ii) Image-finite labelled transition systems are in one-to-one correspondence with coalgebras of the functor ${\mathcal{P}_\omega}^A$.

(iii) Probabilistic transition systems are in one-to-one correspondence with $(1+\mathcal{D})^A$-coalgebras. They can alternatively be modelled as $\mathcal{S}^A$-coalgebras.

(iv) Probabilistic automata are the same as $(\mathcal{P}_\omega \circ \mathcal{D})^A$-coalgebras. They can alternatively be modelled as $(\mathcal{P}_\omega \circ \mathcal{S})^A$-coalgebras.

(v) Spatial transition systems can be modelled as coalgebras of the functor ${\mathcal{P}_\omega}^A \times \mathcal{P}_\omega(\mathsf{Id} \times \mathsf{Id})$.

(vi) Epistemic systems can be modelled as coalgebras of the functor $(1 + \mathsf{Id})^{Ac} \times {\mathcal{P}_\omega}^{Ag} \times \mathcal{P}_\omega(At)$, for some fixed sets Ac of epistemic actions, Ag of agents, and At of atomic facts.

3 Modular Logics for Coalgebras

In this section, we describe a modular approach to defining expressive modal logics for $\mathbb{T}$-coalgebras. We begin with a brief overview of existing work on modal logics for coalgebras.

This direction of work was initiated by Moss [15], who defined a modal logic for coalgebras of an inclusion- and weak-pullback preserving functor $\mathbb{T} : \mathsf{Set} \to \mathsf{Set}$, by using the functor $\mathbb{T}$ itself to derive the syntax of a language, and the lifting $\Gamma_{\mathbb{T}}$ of $\mathbb{T}$ to Rel to provide a coalgebraic semantics for this language. Apart from infinitary conjunctions, the only logical operator in Moss's language is a modal operator, here denoted Δ, whose arity depends on the functor $\mathbb{T}$: if L is a set of formulas of the language and $\Phi \in \mathbb{T}L$, then $\Delta\Phi$ is itself a formula of the language. The semantics of the language thus obtained is defined by structural induction on formulas: having defined a satisfaction relation $\models_\gamma \subseteq C \times L$ for a $\mathbb{T}$-coalgebra (C, γ) and a subset L of the language, the semantics of formulas of form $\Delta\Phi$ is derived using the relation $(\Gamma_{\mathbb{T}} \models_\gamma) \subseteq \mathbb{T}C \times \mathbb{T}L$ together with the coalgebra structure γ; infinitary conjunctions are interpreted in the standard way.

The approach in [15] yields an abstract, infinitary logic, called *coalgebraic logic*, for each endofunctor $\mathbb{T}$. This logic characterises bisimulation, that is, the logical equivalence relation between the sets of states of two $\mathbb{T}$-coalgebras coincides with the bisimilarity relation between the coalgebras. However, for most functors $\mathbb{T}$, this logic is not suitable for use as a specification logic: even for simple functors such as the finite powerset functor, the size of the set of sub-formulas of a given formula grows exponentially with the *rank* of the formula (defined as the maximal degree of nesting of modal operators).

In order to achieve a compromise between uniformity in the functor and suitability for use as a specification logic, several authors have proposed less canonical modal logics for coalgebras, which often still enjoy expressiveness properties similar to those of coalgebraic logic. For instance, Rößiger [17], Kurz [14] and Jacobs [11] have focused on inductively-defined classes of functors, similar to the one considered here but lacking the (sub-)probability distribution functor as a basic coalgebraic type, whereas Pattinson [16] has developed an approach for defining logics for coalgebras of arbitrary functors on Set from specified sets of modal operators.

In [17,11], the structure of the functor $\mathbb{T}$ is exploited in order to define a multi-sorted logic which characterises bisimulation and admits a sound and complete axiomatisation. The sorts of formulas in the modal language of [17,11] correspond to the *ingredients* of $\mathbb{T}$ (the intermediary functors used in the inductive definition of $\mathbb{T}$). As mentioned above, the (sub-)probability distribution functor was not considered in loc. cit.; moreover, this functor can not be straightforwardly incorporated in this approach, since the proofs of the main results in loc. cit. can not be suitably extended to include this functor.

Instead of considering an inductively-defined class of endofunctors, and

deriving the modal operators of the associated language by structural induction on the functor, the approach of Pattinson [16] is to define a logic for $\mathbb{T}$-coalgebras by directly providing a set of unary modal operators, together with sufficient information to interpret these operators over $\mathbb{T}$-coalgebras. *Predicate liftings* are defined in [16] as natural transformations of the form $\lambda : \hat{\mathcal{P}} \Rightarrow \hat{\mathcal{P}}\mathbb{T}$, with $\hat{\mathcal{P}} : \mathsf{Set} \to \mathsf{Set}$ denoting the contravariant powerset functor. Each predicate lifting λ gives rise to a modal operator $[\lambda]$. Writing $[\![\varphi]\!]_C$ for the set of states of a $\mathbb{T}$-coalgebra (C, γ) satisfying the formula φ, the formula $[\lambda]\varphi$ is defined to hold in a state c of the same coalgebra precisely when $\gamma(c) \in \lambda_C([\![\varphi]\!]_C)$.

While the approach in [16] can be applied to arbitrary functors on Set, this approach does not make use of the structure of the underlying functor. This makes it difficult to exhibit a suitable choice of predicate liftings as the functor becomes more complex. Moreover, this approach is not compositional, in that if the functors $\mathbb{T}_1$ and $\mathbb{T}_2$ admit sets of predicate liftings for which the resulting logics characterise bisimulation, this is not necessarily the case for their composition $\mathbb{T}_1 \circ \mathbb{T}_2$ – an example here is the functor $\mathcal{P}_\omega \circ \mathcal{P}_\omega$. This drawback has been overcome in the work of Schröder [19], who shows that by considering a generalisation of predicate liftings giving rise to modal operators of arbitrary (including infinitary) arities, any functor on Set admits a set of such *polyadic* predicate liftings, with the property that the resulting logic is expressive for bisimulation. The problem now is that, in general, an infinite number of modalities, including infinitary modalities, appear to be needed.

The work described in the following represents an alternative approach to deriving modal logics for coalgebras, which subsumes all previously-mentioned approaches. The results presented apply directly to the class of functors defined in (1), but the approach is more general, as it also incorporates the coalgebraic logic of Moss, and logics arising from polyadic predicate liftings. Thus, this approach can be regarded as a unifying framework for modal logics for coalgebras over Set. At the same time, the approach is modular: logics for (coalgebras of) functor combinations can be automatically derived from logics for (coalgebras of) the functors being combined, once a suitable formulation of the logics being put together has been obtained. Furthermore, relevant properties of the resulting logics, including expressiveness and the existence of suitable axiomatisations, can themselves be derived in a modular fashion.

The notions of *syntax constructor* and associated *one-step semantics* [3,6] are central to this approach. They allow the definition of a modal logic for $\mathbb{T}$-coalgebras by specifying a modal syntax (typically a set of modal operators with finite arities), which is subsequently interpreted over $\mathbb{T}$-coalgebras by only carrying out a one-step unfolding of the respective coalgebra maps.

Definition 3.1 A *syntax constructor* is an inclusion-preserving, ω-accessible endofunctor $\mathsf{S} : \mathsf{Set} \to \mathsf{Set}$. The *language* $\mathcal{L}(\mathsf{S})$ *induced by* S is the least set F of formulas such that

- $\mathsf{ff} \in F$,

- $\varphi \to \psi \in F$ whenever $\varphi, \psi \in F$,
- $\sigma \in F$ whenever $\sigma \in \mathsf{S}\Phi$ for some (finite) $\Phi \subseteq F$.

A syntax constructor S specifies the modal operators which need to be added to the basic propositional language in order to obtain a language for $\mathbb{T}$-coalgebras. The language induced by S is then obtained as the least set of formulas which is closed under the application of boolean operators (first two of the above clauses), and of the modal operators specified by S (last clause).

A variation of the notion of language induced by a syntax constructor S can be obtained by choosing a different set of boolean operators. While ff and $\to$ are sufficient to recover all other boolean operators, including negation, conjunction and disjunction, in certain situations (e.g. when looking to logically characterise preorders that are not equivalences) one is interested in a language without negation. In those cases, using tt, ff, $\wedge$ and $\vee$ as a choice of boolean operators is more appropriate. In what follows, and particularly in Section 5, we will also refer to the language $\mathcal{L}_\Sigma(\mathsf{S})$ induced by a syntax constructor S and a set of boolean operators $\Sigma \subseteq \{\mathsf{tt}, \mathsf{ff}, \wedge, \vee, \neg, \to\}$.

The language of standard modal logic can be retrieved by taking $\mathsf{S} : \mathsf{Set} \to \mathsf{Set}$ to be given by $\mathsf{S}L = \{ \Box\varphi \mid \varphi \in L \}$. Similarly, the language of Hennessy-Milner logic can be obtained by taking $\mathsf{S}L = \{ [a]\varphi \mid \varphi \in L,\ a \in A \}$ with A a set (of labels). We also note that the ω-accessibility requirement prevents syntax constructors from specifying modal operators of infinitary arities.

Each of the basic functors in (1) can be associated a syntax constructor in a natural way:

Example 3.2 (i) For $\mathbb{T} := C$, let $\mathsf{S}_C L = C$. The induced language $\mathcal{L}(\mathsf{S}_C)$ is the set of propositional formulas over the set C of atoms.

(ii) for $\mathbb{T} := \mathcal{P}_\omega$, let $\mathsf{S}_{\mathcal{P}_\omega} L = \{ \Box\varphi \mid \varphi \in L \}$. The induced language $\mathcal{L}(\mathsf{S}_{\mathcal{P}_\omega})$ is the language of standard modal logic over an empty set of atoms.

(iii) For $\mathbb{T} := \mathsf{Id}$, let $\mathsf{S}_{\mathsf{Id}} L = \{ \bigcirc\varphi \mid \varphi \in L \}$. The induced language $\mathcal{L}(\mathsf{S}_{\mathsf{Id}})$ is similar to the language of standard modal logic. However, this language will be interpreted over Id-coalgebras (which provide a trivial model of deterministic systems), and will therefore have a different semantics from that of the standard modal language.

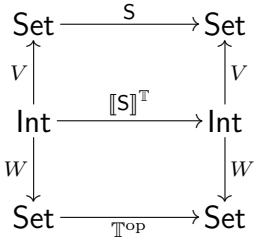
(iv) For $\mathbb{T} = \mathcal{D}$, let $\mathsf{S}_{\mathcal{D}} L = \{ \Diamond_p\varphi \mid \varphi \in L,\ p \in \mathbb{Q} \cap [0,1] \}$. The induced language $\mathcal{L}(\mathsf{S}_{\mathcal{D}})$ is the language commonly used to specify properties of discrete probabilistic systems, including probabilistic transition systems and probabilistic automata (see e.g. [13]); it employs a countable number of unary modalities, with formulas of form $\Diamond_p\varphi$ being read as "the probability of φ holding in the next state is at least p". For $\mathbb{T} = \mathcal{S}$, the same syntax constructor can be used.

A syntax constructor S has yet no direct relationship to a coalgebraic type $\mathbb{T}$. A link between the two is established by providing sufficient information to interpret the induced language $\mathcal{L}(\mathsf{S})$ over $\mathbb{T}$-coalgebras. In the case of $\mathcal{P}_\omega$-

coalgebras, a coalgebraic semantics for $\mathcal{L}(\mathsf{S}_{\mathcal{P}_\omega})$ can be obtained by choosing a suitable predicate lifting to interpret $\Box$. The notion of *one-step semantics* [3,6] generalises this to arbitrary syntax constructors.

If L and X are sets (of formulas and points, respectively), we call a function $d : L \to \mathcal{P}X$ an *interpretation of L over X*. We write Int for the category whose objects are interpretations, and whose morphisms between $d : L \to \mathcal{P}X$ and $d' : L' \to \mathcal{P}X'$ are given by pairs (t, f) with $t : L \to L'$ and $f : X' \to X$ being such that $d' \circ t = f^{-1} \circ d$. Finally, we let $V : \mathsf{Int} \to \mathsf{Set}$ ($W : \mathsf{Int} \to \mathsf{Set}^{\mathrm{op}}$) take $d : L \to \mathcal{P}X$ to L (respectively X), and (t, f) to t (respectively f).

Definition 3.3 A *one-step semantics* $[\![\mathsf{S}]\!]^{\mathbb{T}}$ *for a syntax constructor* S *w.r.t. an endofunctor* $\mathbb{T}$ is a functor $[\![\mathsf{S}]\!]^{\mathbb{T}} : \mathsf{Int} \to \mathsf{Int}$ such that $V \circ [\![\mathsf{S}]\!]^{\mathbb{T}} = \mathsf{S} \circ V$ and $W \circ [\![\mathsf{S}]\!]^{\mathbb{T}} = \mathbb{T}^{\mathrm{op}} \circ W$:



Thus, a one-step semantics $[\![\mathsf{S}]\!]^{\mathbb{T}}$ for S w.r.t. $\mathbb{T}$ maps interpretations of L over X to interpretations of $\mathsf{S}L$ over $\mathbb{T}X$, and moreover, this mapping is functorial. In other words, a one-step semantics describes how to interpret a formula in $\mathsf{S}L$, containing a modal operator at the outer-most level, over $\mathbb{T}X$, provided one has an interpretation of all the sub-formulas to which the modal operator is applied, over X. Some examples of one-step semantics will be given shortly. For simplicity of notation, the superscript in $[\![\mathsf{S}]\!]^{\mathbb{T}}$ will be omitted whenever $\mathbb{T}$ is clear from the context.

A one-step semantics $[\![\mathsf{S}]\!]^{\mathbb{T}}$ for S w.r.t. $\mathbb{T}$ induces a *coalgebraic semantics* for $\mathcal{L}(\mathsf{S})$:

Definition 3.4 The *interpretation* of a formula $\varphi \in \mathcal{L}(\mathsf{S})$ over a $\mathbb{T}$-coalgebra (C, γ), denoted $[\![\varphi]\!]_C \subseteq C$, is defined by structural induction on φ:

- $[\![\mathsf{ff}]\!]_C = \emptyset$
- $[\![\varphi \to \psi]\!]_C = (C \setminus [\![\varphi]\!]_C) \cup [\![\psi]\!]_C$
- $[\![\sigma]\!]_C = \gamma^{-1}([\![\mathsf{S}]\!]^{\mathbb{T}}(d_\Phi)(\sigma))$ for $\sigma \in \mathsf{S}\Phi$

where, for $\Phi \subseteq \mathcal{L}(\mathsf{S})$, $d_\Phi : \Phi \to \mathcal{P}C$ gathers the already-defined interpretations $[\![\varphi]\!]_C$ of formulas $\varphi \in \Phi$. We write $s \models_C \varphi$ whenever $s \in [\![\varphi]\!]_C$.

Each of the syntax constructors in Example 3.2 can be associated a one-step semantics, essentially by choosing a suitable predicate lifting for each of the modal operators used in their definition.

Example 3.5 For $d : L \to \mathcal{P}X$ and $\varphi \in L$, define:

(i) $[\![\mathsf{S}_C]\!](d)(c) = \{c\}$

(ii) $[\![\mathsf{S}_{\mathsf{Id}}]\!](d)(\bigcirc\varphi) = \{x \in X \mid x \in d(\varphi)\}$
(iii) $[\![\mathsf{S}_{\mathcal{P}_\omega}]\!](d)(\Box\varphi) = \{x \in \mathcal{P}_\omega X \mid x \subseteq d(\varphi)\}$
(iv) $[\![\mathsf{S}_{\mathcal{D}}]\!](d)(\Diamond_p\varphi) = \{\mu \in \mathcal{D}X \mid \sum_{x\in d(\varphi)} \mu(x) \geq p\}$

The coalgebraic semantics induced by the above one-step semantics can now be described by:

(i) $s \models_C c$ iff $\gamma(s) = c$
(ii) $s \models_C \bigcirc\varphi$ iff $\gamma(s) \models_C \varphi$
(iii) $s \models_C \Box\varphi$ iff $t \models_C \varphi$ for all $t \in \gamma(s)$
(iv) $s \models_C \Diamond_p\varphi$ iff $\sum_{t\in[\![\varphi]\!]_C} \gamma(s)(t) \geq p$

where s denotes a state of a $\mathbb{T}$-coalgebra (C, γ).

A finitary variant of Moss's coalgebraic logic [15] can also be derived using a suitable choice of syntax constructor and associated one-step semantics.

Example 3.6 If $\mathbb{T}$ is an inclusion-preserving, weak-pullback preserving and ω-accessible endofunctor, then letting $\mathsf{S}_{\mathbb{T}}L = \{\, \Delta\Phi \mid \Phi \in \mathbb{T}L \,\}$ gives rise to a language $\mathcal{L}(\mathsf{S}_{\mathbb{T}})$ whose only modal operator is the modality of coalgebraic logic. Also, letting $[\![\mathsf{S}_{\mathbb{T}}]\!] : \mathsf{Int} \to \mathsf{Int}$ be given by

$$[\![\mathsf{S}_{\mathbb{T}}]\!](d) : \mathsf{S}_{\mathbb{T}}L \to \mathcal{P}\mathbb{T}X \qquad [\![\mathsf{S}_{\mathbb{T}}]\!](d)(\Phi) = \{t \in \mathbb{T}X \mid t\,(\Gamma_{\mathbb{T}} \models_d)\,\Phi\}$$

for $d : L \to \mathcal{P}X$ and $\Phi \in \mathbb{T}L$, where the relation $\models_d \subseteq X \times L$ is given by

$$x \models_d \varphi \text{ iff } x \in d(\varphi)$$

gives rise to a language for $\mathbb{T}$-coalgebras whose syntax and semantics are finitary versions of Moss's coalgebraic logic.

A consequence of the fact that any syntax constructor S is inclusion-preserving and ω-accessible is that the induced language $\mathcal{L}(\mathsf{S})$ can alternatively be given an inductive definition.

Definition 3.7 For $n \in \omega$, the set $\mathcal{L}^n(\mathsf{S})$ of *formulas of rank at most* n is defined inductively by:

- $\mathcal{L}^0(\mathsf{S}) = \mathsf{B}\,\emptyset$,
- $\mathcal{L}^{n+1}(\mathsf{S}) = \mathsf{B}\,\mathsf{S}\mathcal{L}^n(\mathsf{S})$ for $n \in \omega$

where $\mathsf{B} : \mathsf{Set} \to \mathsf{Set}$ takes a set (of atoms) to the carrier of its closure under the boolean operators ff and $\to$.

Proposition 3.8 (Inductive definition of $\mathcal{L}(\mathsf{S})$, [6]) *For a syntax constructor* $\mathsf{S} : \mathsf{Set} \to \mathsf{Set}$, *we have* $\mathcal{L}^n(\mathsf{S}) \subseteq \mathcal{L}^{n+1}(\mathsf{S})$ *and* $\mathcal{L}(\mathsf{S}) = \bigcup_{n\in\omega} \mathcal{L}^n(\mathsf{S})$.

The coalgebraic semantics of $\mathcal{L}(\mathsf{S})$ can itself be given an inductive definition. This is a consequence of the fact that at most n unfoldings of the coalgebra map are required to determine the denotation of a formula of rank

at most n in a $\mathbb{T}$-coalgebra. Following [16], we first interpret formulas of rank n as subsets of $\mathbb{T}^n 1$, with $\mathbb{T}^n$ denoting the n-fold application of $\mathbb{T}$. Specifically, we define interpretations $d_n : \mathcal{L}^n(\mathsf{S}) \to \mathcal{P}\mathbb{T}^n 1$ with $n \in \omega$ by induction on n:

- $d_0 : \mathcal{L}^0(\mathsf{S}) \to \mathcal{P}1$ is the only interpretation that maps ff to $\emptyset$ and $\varphi \to \psi$ to $(1 \setminus d_0(\varphi)) \cup d_0(\psi)$,
- $d_{n+1} : \mathcal{L}^{n+1}(\mathsf{S}) \to \mathcal{P}\mathbb{T}^{n+1}1$ is the natural extension of $[\![\mathsf{S}]\!](d_n)$ to formulas containing boolean operators, for $n \in \omega$.

Now recall that any $\mathbb{T}$-coalgebra (C, γ) induces a cone over the final sequence of $\mathbb{T}$. In particular, one obtains maps of form $\gamma_n : C \to \mathbb{T}^n 1$ with $n \in \omega$.

Proposition 3.9 (Inductive definition of coalgebraic semantics) *The coalgebraic semantics of $\mathcal{L}(\mathsf{S})$ can alternatively be defined by $[\![\varphi]\!]_C = \gamma_n^{-1}(d_n(\varphi))$ for $\varphi \in \mathcal{L}^n(\mathsf{S})$.*

An important property of the coalgebraic semantics of $\mathcal{L}(\mathsf{S})$ is that bisimilar states can not be distinguished by formulas of $\mathcal{L}(\mathsf{S})$. This constitutes an adequacy result for $\mathcal{L}(\mathsf{S})$ w.r.t. $\mathbb{T}$-bisimulation:

Proposition 3.10 (Adequacy of $\mathcal{L}(\mathsf{S})$) *Let (C, γ) and (D, δ) denote $\mathbb{T}$-coalgebras, and let $c \in C$ and $d \in D$. If $c \simeq d$, then $c \models_C \varphi$ iff $d \models_D \varphi$.*

The proof of this result makes use of Proposition 3.9, and of the characterisation of bisimilarity in terms of the relations $\simeq_\alpha$, with α ranging over all ordinals (see Section 2). Proposition 3.9 will be exploited again in the next section, when deriving logical characterisation results w.r.t. (bi)simulations.

Examples 3.2 and 3.5 only account for unlabelled (probabilistic) transition systems, and for two more, rather trivial coalgebraic types; more complex types, similar to the ones of Example 2.1, remain to be dealt with. Since all these more complex types also belong to the inductive class defined in (1), one can attempt to derive syntax constructors and one-step semantics for them in a modular fashion. The remainder of this section is dedicated to this topic.

If L_1, L_2 are sets (of formulas), we define:

$$\begin{aligned} L_1 \otimes L_2 &= \{[\pi_i]\varphi_i \mid \varphi_i \in L_i,\ i = 1, 2\} \\ L_1 \oplus L_2 &= \{\langle \kappa_i \rangle \varphi_i \mid \varphi_i \in L_i,\ i = 1, 2\} \\ L_1 \odot A &= \{[a]\varphi \mid \varphi \in L_1, a \in A\} \end{aligned}$$

where A is an arbitrary set. These operations can be lifted to operations on syntax constructors [3,6], as shown next.

Definition 3.11 For syntax constructors $\mathsf{S}_1, \mathsf{S}_2$, we define

$$\begin{aligned} (\mathsf{S}_1 \otimes \mathsf{S}_2)L &= \mathsf{B}\,\mathsf{S}_1 L \otimes \mathsf{B}\,\mathsf{S}_2 L & (\mathsf{S}_1 \oplus \mathsf{S}_2)L &= \mathsf{B}\,\mathsf{S}_1 L \oplus \mathsf{B}\,\mathsf{S}_2 L \\ (\mathsf{S}_1 \odot A)L &= \mathsf{B}\,\mathsf{S}_1 L \odot A & (\mathsf{S}_1 \circledcirc \mathsf{S}_2)L &= \mathsf{S}_1 \mathsf{B}\,\mathsf{S}_2 L. \end{aligned}$$

The above definition extends naturally to functors $\mathsf{S}_1 \otimes \mathsf{S}_2, \mathsf{S}_1 \odot A, \mathsf{S}_1 \oplus \mathsf{S}_2, \mathsf{S}_1 \circledcirc \mathsf{S}_2 : \mathsf{Set} \to \mathsf{Set}$. Moreover, the resulting operations on functors preserve

the properties of being inclusion-preserving and ω-accessible, and therefore $\mathsf{S}_1 \otimes \mathsf{S}_2, \mathsf{S}_1 \oplus \mathsf{S}_2, \mathsf{S}_1 \odot A, \mathsf{S}_1 \circledcirc \mathsf{S}_2$ also define syntax constructors.

The above combinations of syntax constructors are intended to give rise to modal languages for combinations of coalgebraic types, of the form $\mathbb{T}_1 \times \mathbb{T}_2$, $\mathbb{T}_1 + \mathbb{T}_2$, $\mathbb{T}_1^A$, $\mathbb{T}_1 \circ \mathbb{T}_2$. This is achieved by adding modal operators that mirror the two projections in the case of $\mathbb{T}_1 \times \mathbb{T}_2$, the two injections in the case of $\mathbb{T}_1 + \mathbb{T}_2$, and the exponentiation with constant exponent A in the case of $(\mathbb{T}_1)^A$. No additional modal operator is required for $\mathbb{T}_1 \circ \mathbb{T}_2$. The presence of the functor B in Definition 3.11 is needed to ensure that the resulting languages enjoy logical characterisation results w.r.t. (bi)simulation (see [3,4,6] for details). We also note that the presence of B results in an interleaving of modal operators (either from S_1 or S_2, or of the form $[\pi_i]$, $\langle \kappa_i \rangle$ or $[a]$), with boolean operators. For illustration, we examine the language induced by the composition $\mathsf{S}_1 \circledcirc \mathsf{S}_2$. Suppose $\mathsf{S}_i L = \{\Box_i \varphi \mid \varphi \in L\}$ for $i = 1, 2$. Then, the language $\mathcal{L} = \mathcal{L}(\mathsf{S}_1 \circledcirc \mathsf{S}_2)$ can be described by the following grammar:

$$\begin{array}{ll} \mathcal{L} \ni \varphi, \psi ::= \mathsf{ff} \mid \varphi \to \psi \mid \Box_1 \rho & (\rho \in \mathcal{L}') \\ \mathcal{L}' \ni \rho, \sigma ::= \mathsf{ff} \mid \sigma \to \rho \mid \Box_2 \varphi & (\varphi \in \mathcal{L}) \end{array}$$

Thus, the formulas of $\mathcal{L}(\mathsf{S}_1 \circledcirc \mathsf{S}_2)$ alternate between applications of the modal operators $\Box_1$ and $\Box_2$, and can additionally contain boolean operators at any level. Assuming that S_1 and S_2 specify languages for $\mathbb{T}_1$- and $\mathbb{T}_2$-coalgebras, respectively, the above language can automatically be endowed with a semantics w.r.t. $\mathbb{T}_1 \circ \mathbb{T}_2$-coalgebras. This can be achieved by defining ways to combine a one-step semantics for S_1 w.r.t $\mathbb{T}_1$ with a one-step semantics for S_2 w.r.t $\mathbb{T}_2$, in order to obtain a one-step semantics for $\mathsf{S}_1 \circledcirc \mathsf{S}_2$ w.r.t. $\mathbb{T}_1 \circ \mathbb{T}_2$. More generally, all four operations on syntax constructors have a counterpart at the level of one-step semantics [3,6]:

If $d_1 : L_1 \to \mathcal{P}X_1$ and $d_2 : L_2 \to \mathcal{P}X_2$ are interpretations of L_1 and L_2 over X_1 and X_2, respectively, we define

$$\begin{array}{ll} d_1 \otimes d_2 : L_1 \otimes L_2 \to \mathcal{P}(X_1 \times X_2), & [\pi_i]\varphi_i \mapsto \{(x_1, x_2) \mid x_i \in d_i(\varphi_i)\} \\ d_1 \oplus d_2 : L_1 \oplus L_2 \to \mathcal{P}(X_1 + X_2), & \langle \kappa_i \rangle \varphi_i \mapsto \{\iota_i(x_i) \mid x_i \in d_i(\varphi_i)\} \\ d_1 \odot A : L_1 \odot A \to \mathcal{P}(X^A), & [a]\varphi \mapsto \{f : A \to X \mid f(a) \in d_1(\varphi)\}. \end{array}$$

Definition 3.12 If $[\![\mathsf{S}_i]\!]^{\mathbb{T}_i}$ is a one-step semantics for S_i w.r.t. $\mathbb{T}_i$, for $i = 1, 2$, we define one-step semantics for $\mathsf{S}_1 \otimes \mathsf{S}_2$, $\mathsf{S}_1 \oplus \mathsf{S}_2$, $\mathsf{S}_1 \odot A$, $\mathsf{S}_1 \circledcirc \mathsf{S}_2$ w.r.t. $\mathbb{T}_1 \times \mathbb{T}_2$, $\mathbb{T}_1 + \mathbb{T}_2$, $\mathbb{T}_1^A$, $\mathbb{T}_1 \circ \mathbb{T}_2$, respectively, as follows:

$$\begin{array}{ll} [\![\mathsf{S}_1 \otimes \mathsf{S}_2]\!](d) = [\![\mathsf{S}_1]\!](d)^\sharp \otimes [\![\mathsf{S}_2]\!](d)^\sharp & [\![\mathsf{S}_1 \oplus \mathsf{S}_2]\!](d) = [\![\mathsf{S}_1]\!](d)^\sharp \oplus [\![\mathsf{S}_2]\!](d)^\sharp \\ [\![\mathsf{S}_1 \odot A]\!] = [\![\mathsf{S}_1]\!](d)^\sharp \odot A & [\![\mathsf{S}_1 \circledcirc \mathsf{S}_2]\!](d) = [\![\mathsf{S}_1]\!]([\![\mathsf{S}_2]\!](d)^\sharp) \end{array}$$

where $d : L \to \mathcal{P}X$, and $d^\sharp : \mathsf{B}\, L \to \mathcal{P}X$ denotes the natural extension of d to formulas containing boolean operators.

It follows easily that if $[\![\mathsf{S}_i]\!]$ is a one-step semantics for S_i w.r.t. $\mathbb{T}_i$, for $i = 1, 2$, then $[\![\mathsf{S}_1 \otimes \mathsf{S}_2]\!]$, $[\![\mathsf{S}_1 \oplus \mathsf{S}_2]\!]$, $[\![\mathsf{S}_1 \odot A]\!]$ and $[\![\mathsf{S}_1 \circledcirc \mathsf{S}_2]\!]$ are one-step semantics for $\mathsf{S}_1 \otimes \mathsf{S}_2$, $\mathsf{S}_1 \oplus \mathsf{S}_2$, $\mathsf{S}_1 \odot A$ and $\mathsf{S}_1 \circledcirc \mathsf{S}_2$ w.r.t. $\mathbb{T}_1 \times \mathbb{T}_2$, $\mathbb{T}_1 + \mathbb{T}_2$, ${\mathbb{T}_1}^A$ and $\mathbb{T}_1 \circ \mathbb{T}_2$, respectively.

Example 3.13 The language $\mathcal{L}_1 = \mathcal{L}((\mathsf{S}_{\mathcal{P}_\omega} \circledcirc \mathsf{S}_{\mathcal{D}}) \odot A)$ induced by the combination of syntax constructors $(\mathsf{S}_{\mathcal{P}_\omega} \circledcirc \mathsf{S}_{\mathcal{D}}) \odot A$ can be described by the grammar:

$$\begin{array}{lll} \mathcal{L}_1 \ni \varphi ::= \mathsf{ff} \mid \varphi \to \varphi' \mid [a]\psi & & (\psi \in \mathcal{L}_2) \\ \mathcal{L}_2 \ni \psi ::= \mathsf{ff} \mid \psi \to \psi' \mid \Box\xi & & (\xi \in \mathcal{L}_3) \\ \mathcal{L}_3 \ni \xi ::= \mathsf{ff} \mid \xi \to \xi' \mid \Diamond_p\varphi & & (\varphi \in \mathcal{L}_1) \end{array}$$

A semantics for this language w.r.t. $(\mathcal{P}_\omega \circ \mathcal{D})^A$-coalgebras is automatically obtained as the coalgebraic semantics induced by the combination of one-step semantics $[\![(\mathsf{S}_{\mathcal{P}_\omega} \circledcirc \mathsf{S}_{\mathcal{D}}) \odot A]\!]$. The resulting logic for probabilistic automata is essentially the same as the probabilistic modal logic of [13].

Example 3.14 The language $\mathcal{L}_1 = \mathcal{L}((\mathsf{S}_{\mathcal{P}_\omega} \odot A) \otimes (\mathsf{S}_{\mathcal{P}_\omega} \circledcirc (\mathsf{S}_{\mathsf{Id}} \otimes \mathsf{S}_{\mathsf{Id}})))$ can be described by the grammar:

$$\begin{array}{lll} \mathcal{L}_1 \ni \varphi ::= \mathsf{ff} \mid \varphi \to \varphi' \mid [\pi_1]\psi \mid [\pi_2]\chi & & (\psi \in \mathcal{L}_2, \chi \in \mathcal{L}_4) \\ \mathcal{L}_2 \ni \psi ::= \mathsf{ff} \mid \psi \to \psi' \mid [a]\xi & & (\xi \in \mathcal{L}_3) \\ \mathcal{L}_3 \ni \xi ::= \mathsf{ff} \mid \xi \to \xi' \mid \Box\varphi & & (\varphi \in \mathcal{L}_1) \\ \mathcal{L}_4 \ni \chi ::= \mathsf{ff} \mid \chi \to \chi' \mid \Box\zeta & & (\zeta \in \mathcal{L}_5) \\ \mathcal{L}_5 \ni \zeta ::= \mathsf{ff} \mid \zeta \to \zeta' \mid [\pi_1]\varphi \mid [\pi_2]\varphi & & (\varphi \in \mathcal{L}_1) \end{array}$$

The following grammar defines a sub-language of the above language:

$$\mathcal{L}_1 \ni \varphi ::= \mathsf{ff} \mid \varphi \to \varphi' \mid \boxed{@}\varphi \mid \varphi \parallel \varphi'$$

where we have used the following abbreviations: $\boxed{@}\varphi ::= [\pi_1][a]\Box\varphi$, $\varphi \parallel \varphi' ::= [\pi_2](\neg\Box\neg([\pi_1]\varphi \wedge [\pi_2]\varphi'))$, with boolean negation $\neg$ and boolean conjunction $\wedge$ being defined in the standard way in terms of ff and $\to$. This sub-language involves an A-indexed set of *action modalities* similar to those of Hennessy-Milner logic, as well as a *spatial modality* as found in various spatial logics for concurrency. In particular, we note that the definition of the (binary) spatial modality requires an interleaving between modal operators (of different sorts) and boolean operators, which is not expressible in a language induced by a set of unary predicate liftings. The language $\mathcal{L}((\mathsf{S}_{\mathcal{P}_\omega} \odot A) \otimes (\mathsf{S}_{\mathcal{P}_\omega} \circledcirc (\mathsf{S}_{\mathsf{Id}} \otimes \mathsf{S}_{\mathsf{Id}})))$ can be automatically interpreted over ${\mathcal{P}_\omega}^A \times \mathcal{P}_\omega(\mathsf{Id} \times \mathsf{Id})$-coalgebras (modelling spatial transition systems), using the modular techniques described in this section. The resulting coalgebraic semantics agrees with the standard interpretation of action and spatial modalities. Moreover, since the modal operators $[\pi_i]$ and $[a]$ distribute over all boolean operators, one can show that the sub-language described above is as expressive as the original language $\mathcal{L}_1$.

4 Coalgebraic Simulations

We now describe similar techniques for deriving notions of simulation for the inductive class of coalgebraic types defined in (1). The results presented in this section are taken from [3,4], and build on earlier work on coalgebraic simulations as described in [10,2,12].

Several notions of simulation for coalgebras of a functor $\mathbb{T}$ can be derived by weakening the definition of bisimulation, namely by replacing the lifting $\Gamma_{\mathbb{T}}$ of $\mathbb{T}$ to Rel (as defined in Section 2), with a so-called $\mathbb{T}$-*relator*. We begin by noting that the lifting $\Gamma_{\mathbb{T}} : \mathsf{Rel} \to \mathsf{Rel}$ of a weak-pullback preserving functor $\mathbb{T} : \mathsf{Set} \to \mathsf{Set}$ preserves equality relations and relational composition. The notion of $\mathbb{T}$-relator [10] is obtained by weakening the first condition.

Definition 4.1 Let $\mathbb{T} : \mathsf{Set} \to \mathsf{Set}$. A $\mathbb{T}$-*relator* is an endofunctor $\Gamma : \mathsf{Rel} \to \mathsf{Rel}$ additionally satisfying:

(i) $\mathsf{U} \circ \Gamma = (\mathbb{T} \times \mathbb{T}) \circ \mathsf{U}$; that is, Γ lifts $\mathbb{T}$;

(ii) $\Gamma(=_A) \supseteq =_{\mathbb{T}A}$, where $=_A \subseteq A \times A$ denotes the equality relation on A;

(iii) $\Gamma(S \circ R) = \Gamma(S) \circ \Gamma(R)$ for any $R \subseteq A \times B$ and $S \subseteq B \times C$.

Any $\mathbb{T}$-relator induces a notion of simulation between $\mathbb{T}$-coalgebras [10,12,4]:

Definition 4.2 Let $\Gamma : \mathsf{Rel} \to \mathsf{Rel}$ be a $\mathbb{T}$-relator. A Γ-*simulation* between $\mathbb{T}$-coalgebras (C, γ) and (D, δ) is a Γ-coalgebra having the pair (γ, δ) as a coalgebra map. The largest Γ-simulation between (C, γ) and (D, δ) is called Γ-*similarity* and is denoted $\gtrsim_\Gamma$. If $c \in C$, $d \in D$ are such that $c \gtrsim_\Gamma d$, we say that c *simulates* d.

A Γ-simulation $(R, (\gamma, \delta))$ between (C, γ) and (D, δ) is thus given by a relation $R \subseteq C \times D$ such that $c\, R\, d$ implies $\gamma(c)\ \Gamma(R)\ \delta(d)$ for any $c \in C$ and $d \in D$.

If $\mathbb{T}$ preserves weak pullbacks, then its lifting $\Gamma_{\mathbb{T}}$ to Rel is a $\mathbb{T}$-relator, and moreover, this relator is minimal among all $\mathbb{T}$-relators; that is, $\Gamma_{\mathbb{T}} R \subseteq \Gamma R$ for any $\mathbb{T}$-relator Γ and any relation R (see [10,2] for details). Throughout this section, we assume that the endofunctor $\mathbb{T}$ preserves weak pullbacks.

By considering the minimal $\mathbb{T}$-relator in the definition of simulation, one recovers the notion of $\mathbb{T}$-bisimulation. Then, by weakening the conditions defining the minimal $\mathbb{T}$-relator, one can derive weaker notions of simulation, capturing various notions of refinement between states of $\mathbb{T}$-coalgebras.

Example 4.3 Several notions of simulation for (unlabelled) transition systems can be derived from suitable choices of $\mathcal{P}_\omega$-relators. Here we consider two such choices, namely $\Gamma_\supseteq, \Gamma^R_\supseteq : \mathsf{Rel} \to \mathsf{Rel}$, defined respectively by

$$X\, \Gamma_\supseteq(R)\, Y \quad \text{iff} \quad \forall\, y \in Y\,.\, \exists\, x \in X\,.\, x\, R\, y$$
$$X\, \Gamma^R_\supseteq(R)\, Y \quad \text{iff} \quad X\, \Gamma_\supseteq(R)\, Y \text{ and } (Y = \emptyset \Rightarrow X = \emptyset)$$

with $R \subseteq A \times B$, $X \in \mathcal{P}_\omega A$ and $Y \in \mathcal{P}_\omega B$. Now if $(S, \to)$ and $(T, \to)$ are unlabelled transition systems (i.e. $\mathcal{P}_\omega$-coalgebras), then a $\Gamma_\supseteq$-simulation between them is given by a relation $R \subseteq S \times T$ with the property that whenever $s\, R\, t$ and $t \to t'$ in $(T, \to)$, there exists a transition $s \to s'$ in $(S, \to)$ such that $s'\, R\, t'$. The notion of simulation induced by $\Gamma^R_\supseteq$ additionally requires that if $t \not\to$ in $(T, \to)$, then also the corresponding $s \not\to$ in $(S, \to)$. Thus, the former notion of simulation coincides with standard transition system simulation, whereas the latter captures ready simulation [20].

Example 4.4 In order to define a notion of simulation for (unlabelled) probabilistic transition systems, it is more convenient to model these as $\mathcal{S}$-coalgebras, where $\mathcal{S}$ is the finite sub-probability distribution functor; this allows one not to distinguish between the absence of transitions from a given state, and the existence of transitions from that state, with the associated probabilities adding up to 1. We now define an $\mathcal{S}$-relator $\Gamma_P : \mathsf{Rel} \to \mathsf{Rel}$ by

$$\mu\, (\Gamma_P R)\, \nu \quad \text{iff} \quad \mu[X] \geq \nu[Y] \text{ for any } X \subseteq A \text{ and } Y \subseteq B \text{ s.t.} \\ (\pi_1^R)^{-1}(X) \supseteq (\pi_2^R)^{-1}(Y)$$

with $R \subseteq A \times B$, $\mu \in \mathcal{S}A$ and $\nu \in \mathcal{S}B$. It is shown in [4] that Γ_P is well-defined (as a functor on Rel whose action on arrows is defined via $\mathbb{T}$) and moreover, that Γ_P defines an $\mathcal{S}$-relator. The notion of simulation induced by Γ_P turns out to coincide with the standard notion of simulation for probabilistic transition systems, as defined e.g. in [7] (see [4] for details).

Again, so far we have only considered unlabelled (probabilistic) transition systems. In the following, we show how to derive $\mathbb{T}$-relators , and hence notions of simulation, for functors $\mathbb{T}$ belonging to the inductive class defined in (1).

We begin by defining relators for the other two basic coalgebraic types considered in (1), namely the constant and identity functors. Specifically, for $\mathbb{T} := C$, we let $\Gamma_C : \mathsf{Rel} \to \mathsf{Rel}$ map a relation $R \subseteq A \times B$ to the equality relation on C. Also, for $\mathbb{T} = \mathsf{Id}$, we let Γ_{Id} be the identity functor on Rel. (These relators are in fact the minimal ones.)

Next, we show how to combine a $\mathbb{T}_1$- and a $\mathbb{T}_2$-relator, Γ_1 and Γ_2, in order to obtain a $\mathbb{T}$-relator, with $\mathbb{T}$ being a combination of the functors $\mathbb{T}_1$ and $\mathbb{T}_2$.

Definition 4.5 Let Γ_1 and Γ_2 be $\mathbb{T}_1$- and $\mathbb{T}_2$-relators, respectively. Define $\Gamma_1 \oplus \Gamma_2$, $\Gamma_1 \otimes \Gamma_2$, $(\Gamma_1)^A : \mathsf{Rel} \to \mathsf{Rel}$ by:

- $R \subseteq X \times Y \overset{\Gamma_1 \oplus \Gamma_2}{\longmapsto} \Gamma_1(R) + \Gamma_2(R) \subseteq (\mathbb{T}_1 + \mathbb{T}_2)X \times (\mathbb{T}_1 + \mathbb{T}_2)Y$
- $R \subseteq X \times Y \overset{\Gamma_1 \otimes \Gamma_2}{\longmapsto} \Gamma_1(R) \times \Gamma_2(R) \subseteq (\mathbb{T}_1 \times \mathbb{T}_2)X \times (\mathbb{T}_1 \times \mathbb{T}_2)Y$
- $R \subseteq X \times Y \overset{(\Gamma_1)^A}{\longmapsto} \Gamma_1(R)^A \subseteq (\mathbb{T}_1 X)^A \times (\mathbb{T}_1 Y)^A$

where, given two relations R_1 and R_2, we write $R_1 \times R_2$ and $R_1 + R_2$ for their product and respectively coproduct in Rel, and $(R_1)^A$ for the point-wise extension of R_1 to functions with domain A.

It follows easily (see also [4]) that the above operations yield relators for $\mathbb{T}_1 + \mathbb{T}_2$, $\mathbb{T}_1 \times \mathbb{T}_2$ and $(\mathbb{T}_1)^A$, respectively. Finally, a $\mathbb{T}_1 \circ \mathbb{T}_2$-relator can be obtained from a $\mathbb{T}_1$-relator Γ_1 and a $\mathbb{T}_2$-relator Γ_2 by simply composing them, that is, by considering the $\mathbb{T}_1 \circ \mathbb{T}_2$-relator $\Gamma_1 \circ \Gamma_2$. As a result, we are now able to derive relators for all the coalgebraic types specified in (1).

By combining the relators given in Examples 4.3 and 4.4, one can automatically derive notions of simulation for labelled (probabilistic) transition systems, as well as for more complex types such as probabilistic automata.

Example 4.6 The notions of simulation induced by the ${\mathcal{P}_\omega}^A$-relators $(\Gamma_\supseteq)^A$ and $(\Gamma^R_\supseteq)^A$ coincide with standard, respectively ready simulation on labelled transition systems. The notion of simulation induced by the $\mathcal{S}^A$-relator $(\Gamma_P)^A$ coincides with standard simulation on labelled probabilistic transition systems.

Example 4.7 As mentioned in Example 2.1, probabilistic automata can be modelled as coalgebras of the functor $(\mathcal{P}_\omega \circ \mathcal{S})^A$. Here we derive a notion of simulation for $(\mathcal{P}_\omega \circ \mathcal{S})^A$-coalgebras by combining the $\mathcal{P}_\omega$-relator $\Gamma_\supseteq$ and the $\mathcal{S}$-relator Γ_P. Specifically, we consider the $(\mathcal{P}_\omega \circ \mathcal{S})^A$-relator $(\Gamma_\supseteq \circ \Gamma_P)^A$. A relation $R \subseteq C \times D$ is a $(\Gamma_\supseteq \circ \Gamma_P)^A$-simulation between $(\mathcal{P}_\omega \circ \mathcal{S})^A$-coalgebras (C, γ) and (D, δ) iff $c\,R\,d$ implies:

$$\forall\, a \in A\,.\, \forall\, \nu \in \delta(d)(a)\,.\, \exists\, \mu \in \gamma(c)(a)\,.\, (\,\mu[X] \geq \nu[Y] \text{ whenever } (\pi_1^R)^{-1}(X) \supseteq (\pi_2^R)^{-1}(Y)\,)$$

We will show later that the above notion of simulation coincides with the notion of *strong simulation* on probabilistic automata [13], defined as follows: Given two probabilistic automata $(S, \rightarrow)$ and $(T, \rightarrow)$ (with the transition relations now defining, for each label a, a binary relation between states and probability distributions over states), a *strong simulation* between them is a relation $R \subseteq S \times T$ with the property that whenever $t \xrightarrow{a} \nu$ in $(T, \rightarrow)$, there exists a transition $s \xrightarrow{a} \mu$ in $(S, \rightarrow)$ such that $\mu\,\tilde{R}\,\nu$, where the relation $\tilde{R}$ denotes the *lifting of R to probability distributions* [6] [13]. We conclude by noting that other known notions of simulation for probabilistic automata, including *probabilistic simulation* as defined in [13], can be recovered by using a different choice of $(\mathcal{P}_\omega \circ \mathcal{S})^A$-relator (see [4]).

5 Logical Characterisations

We now proceed to formulating conditions under which the notion of simulation induced by a $\mathbb{T}$-relator Γ can be characterised using the language induced by a syntax constructor S and associated one-step semantics. Since Γ-similarity relations are not, in general, equivalence relations, we will attempt to logically characterise them using languages of the form $\mathcal{L}_\Sigma(\mathsf{S})$, with

[6] This is defined by: $\mu\,\tilde{R}\,\nu$ iff there exists a probability distribution α on $S \times T$ such that $\alpha(s, T) = \mu(s)$ for $s \in S$, $\alpha(S, t) = \nu(t)$ for $t \in T$, and $\alpha(s, t) = 0$ for $(s, t) \notin R$.

$\Sigma \subseteq \{\mathsf{tt}, \mathsf{ff}, \wedge, \vee, \neg, \rightarrow\}$. Throughout this section, Σ will denote a fixed such set of boolean operators, which typically will not include negation, while $\mathsf{B}_\Sigma : \mathsf{Set} \rightarrow \mathsf{Set}$ will denote the functor taking a set (of atoms) to the carrier of its closure under the boolean operators in Σ. Our approach will be based on some well-behavedness properties of Γ-similarity, and on a characterisation of the Γ-similarity relation on the final $\mathbb{T}$-coalgebra, as summarised below.

Proposition 5.1 ([12]) *The following hold for a $\mathbb{T}$-relator $\Gamma : \mathsf{Rel} \rightarrow \mathsf{Rel}$:*

(i) *Γ-similarity on a $\mathbb{T}$-coalgebra (C, γ) is a preorder on C;*

(ii) *given $\mathbb{T}$-coalgebra morphisms $f : (A, \alpha) \rightarrow (B, \beta)$ and $g : (C, \gamma) \rightarrow (D, \delta)$, $a \gtrsim_\Gamma c$ iff $f(a) \gtrsim_\Gamma g(c)$, for $a \in A$ and $c \in C$;*

(iii) *Γ-similarity on the final $\mathbb{T}$-coalgebra is a final Γ-coalgebra.*

By taking f and g in (ii) of Proposition 5.1 to be the unique morphisms $!_\alpha : (A, \alpha) \rightarrow (Z, \zeta)$ and $!_\gamma : (C, \gamma) \rightarrow (Z, \zeta)$ into the final $\mathbb{T}$-coalgebra, we obtain that Γ-similarity between (A, α) and (C, γ) can be derived from the Γ-similarity relation on the final $\mathbb{T}$-coalgebra. Also, the adequacy of logics induced by syntax constructors (Proposition 3.10) results in the satisfaction of formulas being preserved and reflected by coalgebra morphisms. These two observations allow us to restrict attention to logically characterising the Γ-similarity relation on the final $\mathbb{T}$-coalgebra. For this, we make use of (iii) of Proposition 5.1.

We let Preord denote the category of preorders and monotonic maps. Then, Preord is (isomorphic to) a sub-category of Rel, and moreover, any $\mathbb{T}$-relator Γ restricts to an endofunctor on Preord (itself denoted Γ). Motivated by (iii) of Proposition 5.1, we now investigate the final sequence of Γ, which we denote by $(\gtrsim_\Gamma^\alpha)$. It follows easily that this sequence belongs to Preord. Moreover, its underlying Set-sequence is the final sequence of $\mathbb{T}$.

Now recall from Section 3 that a syntax constructor S and choice of one-step semantics for S w.r.t. $\mathbb{T}$ give rise to a sequence of interpretations $d_n : \mathcal{L}_\Sigma^n(\mathsf{S}) \rightarrow \mathcal{P}\mathbb{T}^n 1$, with $n \in \omega$. The sequence $(d_n)_{n \in \omega}$ can be naturally extended to an ordinal-indexed sequence of interpretations (d_α), with $d_\alpha : \mathcal{L}_\Sigma(\mathsf{S}) \rightarrow \mathcal{P}\mathbb{T}^\alpha 1$ for each $\alpha \geq \omega$ [7].

To obtain a logical characterisation of Γ-similarity on the final $\mathbb{T}$-coalgebra, we assume that the final sequence of Γ stabilises at α. Since the final sequence of $\mathbb{T}$ underlies the final sequence of Γ, this sequence must also stabilise at, or before, α. The fact that $\mathcal{L}_\Sigma(\mathsf{S})$ characterises $\gtrsim_\Gamma^\alpha = \gtrsim_\Gamma$ will now follow by induction over the final sequence of Γ, using the notion of *one-step expressiveness* [4] of a one-step semantics w.r.t. a given relator.

If $d : L \rightarrow \mathcal{P}X$ is an interpretation, then for $x, y \in X$, we write $y \geq_L x$ if $x \in d(\varphi)$ implies $y \in d(\varphi)$, for any $\varphi \in L$. Then, d is called *adequate for a*

[7] Note that, while the final sequence of $\mathbb{T}$ might not stabilise at ω, applying S followed by B_Σ to $\mathcal{L}_\Sigma(\mathsf{S})$ does not produce any new formulas. This is the reason for the domains of the interpretations d_α with $\alpha \geq \omega$ being equal to $\mathcal{L}_\Sigma(\mathsf{S})$.

preorder $R \subseteq X \times X$ if $R \subseteq \geq_L$, and *expressive for* R if, in addition, $R \supseteq \geq_L$.

Definition 5.2 A one-step semantics $[\![\mathsf{S}]\!]$ for S w.r.t. $\mathbb{T}$ is called *one-step expressive w.r.t.* Γ if it maps an interpretation $d : L \to \mathcal{P}X$ which is expressive for $R \subseteq X \times X$ to an interpretation $d' : \mathsf{S}L \to \mathcal{P}\mathbb{T}X$ which is expressive for $\Gamma R \subseteq \mathbb{T}X \times \mathbb{T}X$.

One-step expressiveness of $[\![\mathsf{S}]\!]$ w.r.t. Γ ensures that the interpretations d_α are expressive w.r.t. the relations $\gtrsim_\Gamma^\alpha$ in the final sequence of Γ:

Theorem 5.3 ([4]) *Let* $[\![\mathsf{S}]\!] : \mathsf{Int} \to \mathsf{Int}$ *be a one-step semantics for* S *w.r.t.* $\mathbb{T}$. *If* $[\![\mathsf{S}]\!]$ *is one-step expressive w.r.t.* Γ, *then* $d_\alpha : L_\alpha \to \mathcal{P}Z_\alpha$ *is expressive for* $\gtrsim_\alpha \subseteq Z_\alpha \times Z_\alpha$, *for any ordinal* α.

Finally, we are able to formulate sufficient conditions for the language induced by S to characterise Γ-similarity:

Corollary 5.4 (Logical characterisation of simulation, [4]) *Let* $[\![\mathsf{S}]\!] : \mathsf{Int} \to \mathsf{Int}$ *be a one-step semantics for* S *w.r.t.* $\mathbb{T}$. *If* $[\![\mathsf{S}]\!]$ *is one-step expressive w.r.t.* Γ, *and if the final sequence of* Γ *stabilises, then the language* $\mathcal{L}_\Sigma(\mathsf{S})$ *characterises* $\gtrsim_\Gamma$.

The reader might wonder why a stronger requirement on the final sequence of Γ (such as requiring that this sequence stabilises at ω, or at $\omega + \omega$) is not needed for the above result. In fact, from the one-step expressiveness of $[\![\mathsf{S}]\!]$ w.r.t. Γ, and under the additional assumption that $\mathbb{T}$ is ω-accessible, one can prove that the final sequence of Γ stabilises at, or before, $\omega + \omega$ (see [4]). We also note that all the functors defined in (1) are ω-accessible.

The previous result allows us to derive logics which characterise Γ-similarity, from one-step semantics which are one-step expressive w.r.t. Γ. We now derive some concrete logical characterisability results, as instances of Corollary 5.4.

Example 5.5 For unlabelled transition systems, letting $\Sigma = \{\mathsf{tt}, \wedge\}$, $\mathsf{S}^S_{\mathcal{P}_\omega} : \mathsf{Set} \to \mathsf{Set}$ be given by $\mathsf{S}^S_{\mathcal{P}_\omega} L = \{\Diamond\varphi \mid \varphi \in L\}$, and $[\![\mathsf{S}^S_{\mathcal{P}_\omega}]\!] : \mathsf{Int} \to \mathsf{Int}$ be given by $[\![\mathsf{S}^S_{\mathcal{P}_\omega}]\!](d)(\Diamond\varphi) = \{x \in \mathcal{P}_\omega X \mid x \cap d(\varphi) \neq \emptyset\}$ for $d : L \to \mathcal{P}X$ and $\varphi \in L$ yields a one-step semantics for $\mathsf{S}^S_{\mathcal{P}_\omega}$ which is one-step expressive w.r.t. $\Gamma_\supseteq$, and consequently a language $\mathcal{L}_\Sigma(\mathsf{S}^S_{\mathcal{P}_\omega})$ which characterises $\Gamma_\supseteq$-simulation (see [4] for details). In particular, we note that disjunctions are not needed to logically characterise standard simulation on transition systems. To obtain a logical characterisation of ready simulation (on unlabelled transition systems at this point), we enrich the syntax constructor $\mathsf{S}^S_{\mathcal{P}_\omega}$ to $\mathsf{S}^R_{\mathcal{P}_\omega} : \mathsf{Set} \to \mathsf{Set}$ given by $\mathsf{S}^R_{\mathcal{P}_\omega} L = \{\Diamond\varphi \mid \varphi \in L\} \cup \{\Delta\}$; thus, $\mathsf{S}^R_{\mathcal{P}_\omega}$ specifies an additional propositional constant. A one-step semantics $[\![\mathsf{S}^R_{\mathcal{P}_\omega}]\!]$ for $\mathsf{S}^R_{\mathcal{P}_\omega}$ is obtained by letting $[\![\mathsf{S}^R_{\mathcal{P}_\omega}]\!](d)(\Diamond\varphi) = [\![\mathsf{S}^S_{\mathcal{P}_\omega}]\!](d)(\Diamond\varphi)$ and $[\![\mathsf{S}^R_{\mathcal{P}_\omega}]\!](d)(\Delta) = \{\emptyset\}$, for $d : L \to \mathcal{P}X$ and $\varphi \in L$. Again, it is shown in [4] that $[\![\mathsf{S}^R_{\mathcal{P}_\omega}]\!]$ is one-step expressive w.r.t. $\Gamma^R_\supseteq$, and therefore $\mathcal{L}_\Sigma(\mathsf{S}^R_{\mathcal{P}_\omega})$ characterises ready simulation.

Example 5.6 Moving to probabilistic transition systems, and keeping Σ as above, the one-step semantics defined in Section 3 for the syntax constructor

$\mathsf{S}_{\mathcal{D}}$ is one-step expressive w.r.t. the relator Γ_P; as a result, the language induced by $\mathsf{S}_{\mathcal{D}}$ characterises Γ_P-simulation on $\mathcal{S}$-coalgebras (see [4]).

Finally, one expects the one-step expressiveness condition required to derive logical characterisations of simulations to be preserved by the various combinations of one-step semantics and of relators. This is indeed the case:

Theorem 5.7 (Preservation of one-step expressiveness, [4]) *If* $[\![\mathsf{S}_i]\!]$ *is one-step expressive w.r.t.* Γ_i*, for* $i = 1, 2$*, then* $[\![\mathsf{S}_1 \otimes \mathsf{S}_2]\!]$*,* $[\![\mathsf{S}_1 \oplus \mathsf{S}_2]\!]$*,* $[\![\mathsf{S}_1 \odot A]\!]$ *and* $[\![\mathsf{S}_1 \circledcirc \mathsf{S}_2]\!]$ *are one-step expressive w.r.t.* $\Gamma_1 \otimes \Gamma_2$*,* $\Gamma_1 \oplus \Gamma_2$*,* $(\Gamma_1)^A$ *and* $\Gamma_1 \circ \Gamma_2$*, respectively.*

As a result, expressive logics for simulation can be derived in a modular fashion. In particular, one automatically obtains logical characterisations of standard and ready simulation on (image-finite) labelled transition systems, of simulation on probabilistic transition systems, and of $(\Gamma_{\supseteq} \circ \Gamma_P)^A$-simulation on probabilistic automata. We now return to Example 4.7, and note that the notion of strong simulation described there has been shown in [13] to be logically characterisable by essentially the same logic as $\mathcal{L}((\mathsf{S}^S_{\mathcal{P}_\omega} \circledcirc \mathsf{S}_{\mathcal{D}}) \odot A)$ [8]. Since $(\Gamma_{\supseteq} \circ \Gamma_P)^A$-simulation is also characterised by this logic, it follows (indirectly) that $(\Gamma_{\supseteq} \circ \Gamma_P)^A$-simulation coincides with strong simulation on probabilistic automata.

We conclude this section by noting that Hennessy-Milner-style results, providing logical characterisations of $\mathbb{T}$-bisimulation, can be obtained by instantiating the $\mathbb{T}$-relator Γ of Corollary 5.4 with the minimal relator $\Gamma_{\mathbb{T}}$, and appropriately choosing a syntax constructor S, an associated one-step semantics $[\![\mathsf{S}]\!]$ w.r.t. $\mathbb{T}$, and a set of boolean operators Σ. A more direct approach to deriving expressive logics for bisimulation, not involving $\mathbb{T}$-relators, is described in [3,6]. The approach in loc. cit. uses a similar *one-step expressiveness* condition, but this time the definition of expressiveness of an interpretation does not depend on a choice of a $\mathbb{T}$-relator.

6 Sound and Complete Axiomatisations of Coalgebraic Logics

This section describes modular techniques for deriving sound and complete axiomatisations for logics induced by syntax constructors, by summarising the results presented in [6,5]. The section concludes with (part of) a complete axiomatisation for the logic derived earlier for spatial transition systems.

The key idea in defining a proof system for a language of the form $\mathcal{L}(\mathsf{S})$, with S a syntax constructor, is to specify how theorems of rank (at most) $n+1$ can be inferred from already-proved theorems of rank (at most) n.

[8] The logic in [13] is a one-sorted fragment of $\mathcal{L}((\mathsf{S}^S_{\mathcal{P}_\omega} \circledcirc \mathsf{S}_{\mathcal{D}}) \odot A)$, but one can show that it is equally expressive, that is, any formula of $\mathcal{L}((\mathsf{S}^S_{\mathcal{P}_\omega} \circledcirc \mathsf{S}_{\mathcal{D}}) \odot A)$ is semantically equivalent to a formula in this fragment.

This is achieved through the notion of *proof system constructor* [6,5], which typically specifies a set of axioms of rank 1, together with a set of inference rules with premises of rank 0 and conclusion of rank 1.

We use the notion of *boolean theory* to refer to a set of theorems. A boolean theory is defined as a pair (A, Φ_A), with A a set (of atoms) and $\Phi_A \subseteq \mathsf{B}_\Sigma A$ a set (of theorems over A). We write $\vdash \varphi$ for $\varphi \in \Phi_A$ whenever Φ_A is clear from the context, and BTh for the category of boolean theories and morphisms between them (with the latter being given by functions between the corresponding sets of atoms, whose unique extensions to B_Σ-morphisms preserve theorems).

Definition 6.1 A *proof system constructor* for a syntax constructor S is an ω-accessible functor $\mathsf{P} : \mathsf{BTh} \to \mathsf{BTh}$ that satisfies $\mathsf{S} \circ \mathsf{B}_\Sigma \circ \Pi_1 = \Pi_1 \circ \mathsf{P}$, with $\Pi_1 : \mathsf{BTh} \to \mathsf{Set}$ denoting the first projection functor.

A proof system constructor for S lifts the functor $\mathsf{S} \circ \mathsf{B}_\Sigma$, i.e. it maps sets of theorems over A to sets of theorems over $\mathsf{S}\mathsf{B}_\Sigma A$. The requirement that P is ω-accessible generalises a standard requirement in proof systems that inference rules can only contain a finite number of premises. The next example gives proof system constructors for the syntax constructors defined in Example 3.2.

Example 6.2 (i) A proof system constructor $\mathsf{P}_{\mathcal{P}_\omega} : \mathsf{BTh} \to \mathsf{BTh}$ for the syntax constructor $\mathsf{S}_{\mathcal{P}_\omega}$ can be defined by mapping (A, Φ) to $(\mathsf{S}_{\mathcal{P}_\omega}\mathsf{B}_\Sigma A, \Phi')$, where Φ' is generated by the following axioms and rules:

$$\vdash' \Box\mathsf{tt} \qquad \vdash' \Box\varphi \wedge \Box\psi \to \Box(\varphi \wedge \psi) \qquad \frac{\vdash \varphi \to \psi}{\vdash' \Box\varphi \to \Box\psi}$$

$\mathsf{P}_{\mathcal{P}_\omega}$ encodes the axioms and rules of standard modal logic.

(ii) A proof system constructor for $\mathsf{S}_{\mathcal{D}}$ can be defined using a similar, but larger, set of axioms and rules (see [5] for details).

(iii) A proof system constructor $\mathsf{P}_C : \mathsf{BTh} \to \mathsf{BTh}$ for the syntax constructor S_C is given by $\mathsf{P}_C(A, \Phi) = (C, \Phi')$, with Φ' being generated by the axioms:

$$\vdash' \bigvee_{c \in C} c \quad \text{(only if } C \text{ finite)} \qquad \vdash' \neg(c \wedge c') \quad (c \neq c' \in C)$$

(iv) A proof system constructor $\mathsf{P}_{\mathsf{Id}} : \mathsf{BTh} \to \mathsf{BTh}$ for the syntax constructor S_{Id} is given by $\mathsf{P}_{\mathsf{Id}}(A, \Phi) = (\mathsf{S}_{\mathsf{Id}}\mathsf{B}_\Sigma A, \Phi')$, with Φ' being generated by the following axioms and rules:

$$\vdash' \bigcirc\mathsf{ff} \to \mathsf{ff} \qquad \vdash' \bigcirc(\varphi \to \psi) \leftrightarrow (\bigcirc\varphi \to \bigcirc\psi) \qquad \frac{\vdash \varphi \to \psi}{\vdash' \bigcirc\varphi \to \bigcirc\psi}$$

Every proof system constructor P for S induces a boolean theory over $\mathcal{L}(\mathsf{S})$, which contains all the theorems that can be inferred through the application of (the axioms and rules specified by) P, together with the axioms and rules of propositional logic [6,5].

Definition 6.3 The *theory induced by* P is defined as $(\mathcal{L}(\mathsf{S}), \Phi_{\mathsf{P}})$, where Φ_{P} is the least subset Φ of $\mathsf{B}_\Sigma\,\mathcal{L}(\mathsf{S}) = \mathcal{L}(\mathsf{S})$ with the following properties:

- $\mathsf{P}(A, \Psi) \subseteq (\mathcal{L}(\mathsf{S}), \Phi)$ for any $(A, \Psi) \subseteq (\mathcal{L}(\mathsf{S}), \Phi)$ with A and Ψ finite,
- $(\mathcal{L}(\mathsf{S}), \Phi)$ contains all instances of propositional tautologies, and is closed under modus ponens.

We write $\vdash_{\mathsf{P}} \varphi$ for $\varphi \in \Phi_{\mathsf{P}}$.

As in the case of syntax constructors, the ω-accessibility requirement on a proof system constructor P results in an alternative inductive definition of the theory induced by P. This involves defining an ω-indexed set of boolean theories $(\mathcal{A}^n(\mathsf{S}), \Phi^n_{\mathsf{P}})_{n\in\omega}$, with $(\mathcal{A}^0(\mathsf{S}), \Phi^0_{\mathsf{P}})$ being the closure of the empty theory over an empty set of atoms under instances of tautologies and modus ponens, and with $(\mathcal{A}^{n+1}(\mathsf{S}), \Phi^{n+1}_{\mathsf{P}})$ being obtained by applying P to $(\mathcal{A}^n(\mathsf{S}), \Phi^n_{\mathsf{P}})$ and subsequently closing the resulting boolean theory under instances of tautologies and modus ponens. Details can be found in [5].

A consequence of the inductive definition of $(\mathcal{L}(\mathsf{S}), \Phi_{\mathsf{P}})$ is the availability of induction for proving properties (e.g. soundness and completeness) of $(\mathcal{L}(\mathsf{S}), \Phi_{\mathsf{P}})$. Similarly to our approach to deriving logical characterisations of simulation relations, we define notions of *one-step soundness* and *one-step completeness* of a proof system constructor w.r.t. a one-step semantics, and use them to prove soundness and completeness of the induced boolean theory w.r.t. the coalgebraic semantics of $\mathcal{L}(\mathsf{S})$.

Given a boolean theory $(A, \vdash)$, we write $\mathsf{Cl}(A, \vdash)$ for the boolean theory obtained by adding all propositional tautologies over A to $\vdash$, and subsequently closing the resulting set of formulas under modus ponens. We call a boolean theory $(A, \vdash)$ *sound* (*complete*) w.r.t. an interpretation $d : A \to \mathcal{P}X$ if $\vdash \varphi$ implies $d^\sharp(\varphi) = X$ (respectively $d^\sharp(\varphi) = X$ implies $\vdash \varphi$) for any $\varphi \in \mathsf{B}_\Sigma\, A$.

Definition 6.4 A proof system constructor P for S is *one-step sound* (*one-step complete*) w.r.t. a one-step semantics $[\![\mathsf{S}]\!]^{\mathbb{T}}$ if $(\mathsf{Cl} \circ \mathsf{P})(A, \vdash)$ is sound (complete) w.r.t. $[\![\mathsf{S}]\!]^{\mathbb{T}}(d^\sharp) : \mathsf{S}\mathsf{B}_\Sigma\, A \to \mathcal{P}\mathbb{T}X$ whenever $(A, \vdash)$ is sound (respectively complete) w.r.t. $d : A \to \mathcal{P}X$.

Theorem 6.5 (Soundness and completeness, [5]) *If the proof system constructor* P *for* S *is one-step sound (complete) w.r.t.* $[\![\mathsf{S}]\!]^{\mathbb{T}}$*, then* $(\mathcal{L}(\mathsf{S}), \vdash_{\mathsf{P}})$ *is sound (respectively complete) w.r.t. the coalgebraic semantics of* $\mathcal{L}(\mathsf{S})$*, that is,* $\models_{\mathbb{T}} \varphi$ *iff* $\vdash_{\mathsf{P}} \varphi$ *for all* $\varphi \in \mathcal{L}(\mathsf{S})$ *(where* $\models_{\mathbb{T}} \varphi$ *stands for* $c \models_C \varphi$ *for any* $\mathbb{T}$*-coalgebra* (C, γ) *and any* $c \in C$*).*

As shown in [5], each of the proof system constructors in Example 6.2 is one-step sound and complete. The proofs are straightforward, except for the case of the probability distribution functor, where a version of the *theorem of the alternative* for vector spaces is used, following an existing completeness proof in [9]. (A complete proof is given in [5].)

Finally, we show that proof system constructors for different coalgebraic

types can be combined, and that these combinations preserve one-step soundness and one-step completeness. As in previous sections, we define operations $\_ \otimes \_$, $\_ \oplus \_$, $\_ \odot A$, $\_ \circledcirc \_$ on proof system constructors, which lift the corresponding operations on syntax constructors. In the case of the first three operations, additional axioms and rules, axiomatising cartesian products, coproducts and exponents, are required in order to derive completeness results. For example, in the case of products, assuming that P_1 and P_2 are proof system constructors for S_1 and S_2, a proof system constructor $\mathsf{P}_1 \otimes \mathsf{P}_2$ for $\mathsf{S}_1 \otimes \mathsf{S}_2$ is defined by:

$$(\mathsf{P}_1 \otimes \mathsf{P}_2)(A, \vdash) \;=\; (\mathsf{Cl} \circ \mathsf{P}_1)(A, \vdash) \otimes (\mathsf{Cl} \circ \mathsf{P}_2)(A, \vdash)$$

where the operation $\_ \otimes \_$ on proof systems is defined by

$$(A_1, \vdash_1) \otimes (A_2, \vdash_2) = (\mathsf{B}_\Sigma A_1 \otimes \mathsf{B}_\Sigma A_2, \vdash_\otimes)$$

with $\vdash_\otimes$ being generated by the following axioms and rules:

$$\vdash_\otimes [\pi_i]\mathsf{ff} \to \mathsf{ff} \qquad \vdash_\otimes [\pi_i](\varphi \to \psi) \leftrightarrow ([\pi_i]\varphi \to [\pi_i]\psi) \qquad \frac{\vdash_i \varphi \to \psi}{\vdash_\otimes [\pi_i]\varphi \to [\pi_i]\psi}$$

The definition of $\mathsf{P}_1 \odot A$ is similar to that of $\mathsf{P}_1 \otimes \mathsf{P}_2$ – the modal operators $[a]$ with $a \in A$ have similar properties to those of $[\pi_i]$, whereas the definition of $\mathsf{P}_1 \oplus \mathsf{P}_2$ is given in terms of the dual modalities $[\kappa_i]$ of $\langle \kappa_i \rangle$, and includes axioms capturing the distributivity of $[\kappa_i]$ over conjunctions and non-empty disjunctions, some additional properties of coproducts, and an inference rule similar to the one for $\mathsf{P}_1 \otimes \mathsf{P}_2$. Finally, as $\mathsf{P}_1 \circledcirc \mathsf{P}_2$ one can simply consider $\mathsf{P}_1 \circ \mathsf{Cl} \circ \mathsf{P}_2$, similarly to the definition of $\circledcirc$ on syntax constructors.

Theorem 6.6 (Preservation of one-step completeness, [5]) *If P_i is a proof system constructor for S_i, for $i = 1, 2$, then $\mathsf{P}_1 \otimes \mathsf{P}_2$, $\mathsf{P}_1 \oplus \mathsf{P}_2$, $\mathsf{P}_1 \odot A$ and $\mathsf{P}_1 \circledcirc \mathsf{P}_2$ are proof system constructors for $\mathsf{S}_1 \otimes \mathsf{S}_2$, $\mathsf{S}_1 \oplus \mathsf{S}_2$, $\mathsf{S}_1 \odot A$ and $\mathsf{S}_1 \circledcirc \mathsf{S}_2$, respectively. Moreover, if P_1 and P_2 are one-step sound (complete) w.r.t. $[\![\mathsf{S}_1]\!]$ and $[\![\mathsf{S}_2]\!]$, respectively, then $\mathsf{P}_1 \otimes \mathsf{P}_2$, $\mathsf{P}_1 \oplus \mathsf{P}_2$, $\mathsf{P}_1 \odot A$ and $\mathsf{P}_1 \circledcirc \mathsf{P}_2$ are one-step sound (respectively complete) w.r.t. $[\![\mathsf{S}_1 \otimes \mathsf{S}_2]\!]$, $[\![\mathsf{S}_1 \oplus \mathsf{S}_2]\!]$, $[\![\mathsf{S}_1 \odot A]\!]$ and $[\![\mathsf{S}_1 \circledcirc \mathsf{S}_2]\!]$, respectively.*

Theorem 6.6 together with the earlier observation that all the basic proof system constructors are one-step sound and complete yield sound and complete axiomatisations for all the coalgebraic types defined in (1), including probabilistic automata and spatial transition systems. A complete axiomatisation for the language $\mathcal{L}((\mathsf{S}_{\mathcal{P}_\omega} \circledcirc \mathsf{S}_{\mathcal{D}}) \odot A)$, as described in Example 3.13, is given in [5]. Below we give part of the complete axiomatisation obtained for the language $\mathcal{L}((\mathsf{S}_{\mathcal{P}_\omega} \odot A) \otimes (\mathsf{S}_{\mathcal{P}_\omega} \circledcirc (\mathsf{S}_{\mathsf{Id}} \otimes \mathsf{S}_{\mathsf{Id}})))$ described in Example 3.14:

- Axioms and rules for all $\vdash_i$:

$$\vdash_i \varphi \quad (\varphi \text{ instance of tautology}) \qquad \frac{\vdash_i \varphi \quad \vdash_i \varphi \to \psi}{\vdash_i \psi}$$

- Axioms and rules for $\vdash_1$ ($\psi, \psi' \in \mathcal{L}_2$, $\chi, \chi' \in \mathcal{L}_4$):

$$\vdash_1 [\pi_1]\mathsf{tt} \qquad \vdash_1 [\pi_1]\psi \wedge [\pi_1]\psi' \to [\pi_1](\psi \wedge \psi') \qquad \frac{\vdash_2 \psi \to \psi'}{\vdash_1 [\pi_1]\psi \to [\pi_1]\psi'}$$

$$\vdash_1 [\pi_2]\mathsf{tt} \qquad \vdash_1 [\pi_2]\chi \wedge [\pi_2]\chi' \to [\pi_2](\chi \wedge \chi') \qquad \frac{\vdash_4 \chi \to \chi'}{\vdash_1 [\pi_2]\chi \to [\pi_2]\chi'}$$

- Axioms and rules for $\vdash_3$ ($\varphi, \varphi' \in \mathcal{L}_1$):

$$\vdash_3 \Box\mathsf{tt} \qquad \vdash_3 \Box\varphi \wedge \Box\varphi' \to \Box(\varphi \wedge \varphi') \qquad \frac{\vdash_1 \varphi \to \varphi'}{\vdash_3 \Box\varphi \to \Box\varphi'}$$

7 Concluding Remarks

This paper has focused on an inductively-defined class of coalgebraic types, which subsumes many types of interest in the modelling of state-based systems, as it accounts for combinations of (non-)deterministic and probabilistic behaviour, as well as for spatial and epistemic aspects of systems. The techniques described here allow the automatic derivation of modal logics, notions of simulation, logical characterisations, and sound and complete axiomatisations for each of these coalgebraic model types. Many of the results formulated still hold when the finite powerset functor in (1) is replaced with its unbounded version (see [5]) – the restriction to finite powersets (and image-finite transition systems) is only required to logically characterise (bi)simulation relations.

Other basic coalgebraic types such as the *list functor* (mapping a set to the set of lists with elements from that set), as well as further combinations of coalgebraic types, e.g. arising from categorical constructs such as pullbacks or pushouts, could also be added to the inductive class defined in (1).

Ongoing work includes (i) extending the results on modularly deriving modal logics to temporal logics for coalgebras, and (ii) developing modular model-based verification methodologies for coalgebraic models.

References

[1] P. Aczel and N. Mendler. A final coalgebra theorem. In D. H. Pitt et al, editor, *Category Theory and Computer Science*, volume 389 of *Lecture Notes in Computer Science*. Springer, 1989.

[2] A. Baltag. A logic for coalgebraic simulation. In H. Reichel, editor, *Coalgebraic Methods in Computer Science*, volume 33 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2000.

[3] C. Cîrstea. A compositional approach to defining logics for coalgebras. *Theoretical Computer Science*, 327:45–69, 2004.

[4] C. Cîrstea. A modular approach to defining and characterising notions of simulation. *Information and Computation*, 204(4):468–502, 2006.

[5] C. Cîrstea and D. Pattinson. Modular construction of complete coalgebraic logics. Draft available from http://www.ecs.soton.ac.uk/~cc2/.

[6] C. Cîrstea and D. Pattinson. Modular construction of modal logics. In *Proceedings of CONCUR 2004*, volume 3170 of *Lecture Notes in Computer Science*, pages 258–275. Springer, 2004.

[7] J. Desharnais. Logical characterisation of simulation for Markov chains. In *Proceedings of PROBMIV'99*, pages 33–48. University of Birmingham, 1999.

[8] H.P. Gumm and T. Schröder. Types and coalgebraic structure. *Algebra Universalis*, 53:229–252, 2005.

[9] A. Heifetz and P. Mongin. Probability logic for type spaces. *Games and Economic Behaviour*, 35, 2001.

[10] W.H. Hesselink and A. Thijs. Fixpoint semantics and simulation. *Theoretical Computer Science*, 238:275–311, 2000.

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

[12] B. Jacobs and J. Hughes. Simulations in coalgebra. In H.P. Gumm, editor, *Coalgebraic Methods in Computer Science*, volume 82.1 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2003.

[13] B. Jonsson, K.G. Larsen, and W. Yi. Probabilistic extensions of process algebras. In J.A. Bergstra et al, editor, *Handbook of Process Algebra*, pages 685–710. Elsevier, 2001.

[14] A. Kurz. Specifying coalgebras with modal logic. *Theoretical Computer Science*, 260:119–138, 2001.

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

[16] D. Pattinson. Expressive logics for coalgebras via terminal sequence induction. *Notre Dame Journal of Formal Logic*, 45(1):19–33, 2004.

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

[18] J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000.

[19] L. Schröder. Expressivity of coalgebraic modal logic: The limits and beyond. In *Proceedings of FOSSACS 2005*, volume 3441 of *Lecture Notes in COmputer Science*, pages 440–454. Springer, 2004.

[20] R.J. van Glabbeek. The linear time – branching time spectrum I; the semantics of concrete, sequential processes. In J.A. Bergstra et al, editor, *Handbook of Process Algebra*, chapter 1, pages 3–99. Elsevier, 2001.

[21] J. Worrell. On the final sequence of a finitary set functor. *Theoretical Computer Science*, 338(184–199), 2005.