Predicate Liftings and Functor Presentations in Coalgebraic Expression Languages

Ulrich Dorsch(✉), Stefan Milius, Lutz Schröder, and Thorsten Wißmann

Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
{ulrich.dorsch,stefan.milius,lutz.schroeder,thorsten.wissmann}@fau.de

Abstract. We introduce a generic expression language describing behaviours of finite coalgebras over sets; besides relational systems, this covers, e.g., weighted, probabilistic, and neighbourhood-based system types. We prove a generic Kleene-type theorem establishing a correspondence between our expressions and finite systems. Our expression language is similar to one introduced in previous work by Myers but has a semantics defined in terms of a particular form of predicate liftings as used in coalgebraic modal logic; in fact, our expressions can be regarded as a particular type of modal fixed point formulas. The predicate liftings in question are required to satisfy a natural preservation property; we show that this property holds in particular for the Moss liftings introduced by Marti and Venema in work on lax extensions.

1 Introduction

Expression languages that support the syntactic description of system behaviour are one of the classical topics in computer science. The prototypic example are regular expressions; further examples include Kleene algebra with tests [17] and expression languages for labelled transition systems [1].

There has been recent interest in phrasing such expression languages generically, obtaining their syntax and semantics as well as meta-theoretic results including Kleene theorems by instantiation of a parametrized framework. This is achieved by abstracting the type of systems as coalgebras for a given type functor. This line of work originates with expression languages for a specific class of functors that essentially covers relational systems, so-called *Kripke polynomial functors* [34], and was subsequently extended to cover also weighted systems [32]. A generic expression language for arbitrary finitary functors can be based on algebraic functor presentations [25]. Here, we introduce a similar and, as it will turn out, in fact largely equivalent generic expression language

Work forms part of the DFG project COAX (MI 717/5-1/SCHR 1118/11-1).

C. Cîrstea (Ed.): CMCS 2018, LNCS 11202, pp. 56–77, 2018.
https://doi.org/10.1007/978-3-030-00389-0_5

for finitary functors, which we base on coalgebraic modalities in predicate lifting style, following the paradigm of coalgebraic logic [9]; on predicate liftings, we impose strong conditions, notably including preservation of singletons. Marti and Venema [20] have shown that for functors admitting a *lax extension* (in particular for functors that admit a separating set of monotone predicate liftings), one can convert operations from the functor presentation into predicate liftings, the so-called *Moss liftings*. We show that the Moss liftings preserve singletons; the converse does not hold in general, i.e. not all singleton-preserving predicate liftings are Moss liftings under a given lax extension.

We thus arrive at a generic expression language that covers, e.g., various flavours of relational, weighted, and probabilistic systems, as well as monotone neighbourhood systems as in the semantics of game logic [26] and concurrent dynamic logic [29]. We prove a Kleene theorem stating that every expression denotes the behavioural equivalence class of some state in a finite system, and that conversely every such behavioural equivalence class is denoted by some expression.

We make no claim to novelty for the design of a generic expression language as such, and in fact the expression language developed by Myers in his PhD dissertation [25] appears to be even more general. In particular, unlike Myers' language our expression language is currently restricted to describing behavioural equivalence classes in set-based coalgebras, and does not yet support algebraic operations (e.g. a join semilattice structure as in Silva et al.'s language for Kripke-polynomial functors [34] or in fact in standard regular expressions). The main point we are making is, in fact, a different one: we show that

coalgebraic expression languages embed into coalgebraic logic,

specifically into (the conjunctive fragment of) the coalgebraic μ-calculus [8], extending the classical result that every bisimilarity class of states in finite labelled transition systems is expressible by a *characteristic formula* in the μ-calculus [2,10,14,35]. This result provides a direct link between descriptions of processes and their property-oriented specification; as indicated above, the key to lifting it to a coalgebraic level of generality are singleton-preserving predicate liftings.

Related Work. As mentioned above, we owe much to work by Marti and Venema on Moss liftings [20], and moreover we use a notion of *Λ-bisimulation* [12] that turns out to be an instance of their definition of bisimulation via lax extensions. Besides the mentioned work on generic expression languages for Kripke polynomial [34], weighted [32], and finitary [25] functors, there is work on expression languages for *reactive T-automata* [11], which introduce an orthogonal dimension of genericity: The coalgebra functor as such remains fixed but the computational capacities of the automaton model at hand are encapsulated as a computational monad [23]. Venema [38] proves that for weak-pullback preserving fuctors, every bisimilarity class of finite coalgebras is expressible in coalgebraic fixpoint logic over Moss' ∇ modality.

2 Preliminaries

In the standard paradigm of universal coalgebra, types of state-based systems are encapsulated as endofunctors. We recall details on presentations of set functors and on their property-oriented description via predicate-lifting based coalgebraic modalities.

Functor Presentations describe set functors by signatures of operations and a certain restricted form of equations, so-called *flat* equations, alternatively by a suitable natural surjection. A *signature* is a sequence $\Sigma = (\Sigma_n)_{n\in\omega}$ of sets. Elements of Σ_n are regarded as n-ary operation symbols (we write $\tau/n \in \Sigma$ for $\tau \in \Sigma_n$). Every signature Σ determines the corresponding polynomial endofunctor T_Σ on **Set**, which maps a set X to the set

$$T_\Sigma X = \coprod_{n\in\omega} \Sigma_n \times X^n$$

and similarly on maps.

Definition 2.1. A *presentation* of a functor $T : \mathbf{Set} \to \mathbf{Set}$ is a pair (Σ, α) consisting of a signature Σ and a natural transformation $\alpha : T_\Sigma \twoheadrightarrow T$ with surjective components α_X. In the following, we abuse notation and denote, for every $\tau/n \in \Sigma$, the corresponding coproduct component of $\alpha : T_\Sigma \twoheadrightarrow T$ again by $\tau : (-)^n \to T$, and refer to it as an *operation* of T.

Most of our results concern finitary set functors. Recall that a functor is *finitary* if it preserves filtered colimits. Over **Set**, we have the following equivalent characterizations:

Theorem 2.2 (Adámek and Trnkova [3]). *Let $T : \mathbf{Set} \to \mathbf{Set}$ be a functor. Then the following are equivalent:*

1. *T is finitary;*
2. *T is* bounded*, i.e. for every element $x \in TX$ there exists a finite subset $m : Y \hookrightarrow X$ and an element $y \in TY$ such that $x = Tm(y)$;*
3. *T has a presentation.*

Indeed, for the equivalence of (1) and (3) note that every polynomial functor T_Σ is finitary, and finitary functors are closed under taking quotient functors. Conversely, given a finitary functor T, let $\Sigma_n = Tn$ and define $\alpha_X : T_\Sigma X \to TX$ by $\alpha_X(\tau, t) = Tt(\tau)$, where $t \in X^n$ is considered as a function $n \to X$. It is easy to show that this yields a natural transformation with surjective components.

Remark 2.3. As indicated above, the natural surjection α in a functor presentation (Σ, α) can be replaced with a set of flat equations over Σ, where an equation is called *flat* if both sides consist of an operation symbol applied to variables [3]. Incidentally, this (standard) term should not be confused with the same term introduced in the context of our expression language in Sect. 5.

Example 2.4. (1) Let A be an input alphabet. The functor $TX = 2 \times X^A$, whose coalgebras are deterministic automata, is polynomial, and finitary if A is finite. Thus, T has a presentation (Σ, α) by a signature Σ with two $|A|$-ary operations and no equations, i.e. α is the natural isomorphism $T_\Sigma \cong 2 \times (-)^A$.

(2) For a commutative monoid $(M, +, 0_M)$ the monoid-valued functor $M^{(-)} : \mathbf{Set} \to \mathbf{Set}$ is defined by

$$M^{(X)} = \{\mu : X \to M \mid \mu(x) = 0_M \text{ for all but finitely many } \ x \in X\}$$

and by $M^{(h)}(\mu) = y \mapsto \sum_{h(x)=y} \mu(x)$ on maps $h : X \to Y$. We view elements of $M^{(X)}$ as finitely supported additive measures on X, and in particular write $\mu(A) = \sum_{x \in A} \mu(x)$ for $A \subseteq X$; in this view, maps $M^{(h)}$ just take image measures. For a set $G \subseteq M$ of generators (i.e. there exists a surjective monoid morphism $G^* \twoheadrightarrow M$), $M^{(-)}$ is represented by

$$\alpha_X : \coprod_{n \in \omega} G^n \times X^n \twoheadrightarrow M^{(X)}, \qquad \alpha_X(\tau, t) = M^{(t)}(\tau),$$

where $\tau \in G^n$ is considered as an element of $M^{(n)}$.

(3) The finite powerset functor $\mathcal{P}_\omega$ (with $\mathcal{P}_\omega(X)$ being the set of finite subsets of X) is the monoid-valued functor for the monoid $(\{0, 1\}, \vee, 0)$. Since this is generated by $G = \{1\}$, we have one n-ary operation symbol for each $n \in \omega$:

$$\alpha_X : \coprod_{n \in \omega} X^n \twoheadrightarrow \mathcal{P}_\omega X, \qquad \alpha_X(x_1, \ldots, x_n) = \{x_1, \ldots x_n\};$$

e.g. α identifies the tuples (x_1, x_1, x_2) and (x_1, x_2).

(4) For the monoid $\mathbb{N}$ of natural numbers with addition, one obtains the bag functor $\mathcal{B}$ as $\mathbb{N}^{(-)}$. Concretely, $\mathcal{B}$ maps a set X to the set $\mathcal{B}X$ of bags (i.e. finite multisets) on X. Since $(\mathbb{N}, +, 0)$ is generated by $G = \{1\}$, we have the same signature as for $\mathcal{P}_\omega$, namely one n-ary operation symbol per $n \in \omega$; of course, the presentation α now identifies fewer tuples, e.g. distinguishes (x_1, x_2, x_1) and (x_1, x_2).

(5) The *finite distribution functor* $\mathcal{D}$ is a subfunctor of the monoid-valued functor $\mathbb{R}_{\geq 0}^{(-)}$ for the additive monoid of the non-negative reals, given by $\mathcal{D}X = \{\mu \in \mathbb{R}_{\geq 0}^{(-)} \mid \sum_{x \in X} \mu(x) = 1\}$. Note that elements of $\mathcal{D}X$ can be represented as formal convex combinations $\sum_{i=1}^n p_i x_i$, $p_i \in \mathbb{R}_{\geq 0}, x_i \in X$ for $i = 1, \ldots, n$, with $p_1 + \cdots + p_n = 1$. Taking $\mathbb{R}_{\geq 0}$ itself as the set of generators and restricting to $\mathcal{D}$, we obtain a presentation (Σ, α) with an n-ary operation symbol for each n-tuple $(p_1, \ldots, p_n) \in \mathbb{R}_{\geq 0}^n$ such that $p_1 + \cdots + p_n = 1$, and α_X maps $((p_1, \ldots, p_n), (x_1, \ldots, x_n))$ to the formal convex combination $\sum_{i=1}^n p_i x_i$.

(6) The *finitary monotone neighbourhood functor* $\mathcal{M}_\omega$, i.e. the finitary part of the standard monotone neighbourhood functor $\mathcal{M}$, can be described as follows. To begin, $\mathcal{M}$ is the subfunctor of the double contravariant powerset functor $\mathcal{Q}\mathcal{Q}^{\mathsf{op}}$ given on objects by

$$\mathcal{M}X = \{\mathfrak{A} \subseteq \mathcal{Q}(X) \mid \mathfrak{A} \text{ upwards closed under } \subseteq\}.$$

We can then describe $\mathcal{M}_\omega X$ as consisting of all $\mathfrak{A} \in \mathcal{M}X$ having finitely many minimal elements, all of them finite, such that every element of $\mathfrak{A}$ is above a minimal one. We have the following presentation of $\mathcal{M}_\omega$: For every choice of numbers $n \geq 0$, $k_1, \ldots, k_n \geq 0$, we have a $\sum_{i=1}^{n} k_i$-ary operation mapping $(x_{ij})_{i=1,\ldots,n;j=1,\ldots,k_i}$ to the upwards closure of the set system

$$\{\{x_{i1}, \ldots, x_{ik_i}\} \mid i = 1, \ldots, n\}.$$

Coalgebraic Logic. Since coalgebras serve as generic models of reactive systems, it is natural to specify properties of coalgebras in terms of suitable modalities. The semantics of coalgebraic modalities can be defined using *predicate liftings* [27,30], which specify how a predicate on a base set X induces a predicate on the set TX where T is the coalgebraic type functor:

Definition 2.5. For $n \in \omega$ an n-ary *predicate lifting* for a functor $T : \mathbf{Set} \to \mathbf{Set}$ is a natural transformation

$$\lambda : \mathcal{Q}^n \to \mathcal{Q}T^{\mathsf{op}}$$

where $\mathcal{Q} : \mathbf{Set}^{\mathsf{op}} \to \mathbf{Set}$ is the contravariant powerset functor, with $\mathcal{Q}f$ taking preimages, i.e.

$$\mathcal{Q}f(A) = f^{-1}[A].$$

We write λ/n to indicate that λ has arity n. A predicate lifting λ is *monotone* if it preserves set inclusion in every argument. A set Λ of predicate liftings is *separating* [28,30] if every $t \in TX$ is uniquely determined by the set

$$\mathcal{T}_\Lambda(t) = \{(\lambda, A_1, \ldots, A_n) \mid \lambda/n \in \Lambda, A_i \in \mathcal{Q}X \text{ and } t \in \lambda_X(A_1, \ldots, A_n)\}.$$

Example 2.6. The basic example is the interpretation of the standard box modality $\Box$ over the covariant powerset functor $\mathcal{P}$ (with $\mathcal{P}f$ taking direct images), given by the monotone unary predicate lifting λ defined by

$$\lambda_X(A) = \{B \in \mathcal{P}(X) \mid B \subseteq A\}.$$

For a further monotone example, we interpret the box modality over the monotone neighbourhood functor $\mathcal{M}$ (Example 2.4) by the monotone unary predicate lifting

$$\lambda_X(A) = \{\mathfrak{A} \in \mathcal{M}X \mid A \in \mathfrak{A}\}.$$

It is easy to see that in both these examples, the predicate lifting for $\Box$ alone is separating.

Predicate-lifting-based modalities can be embedded into *coalgebraic logics* of varying degrees of expressiveness. Our expression language introduced in Sect. 5 will live inside the *coalgebraic μ-calculus* [8], more precisely its *conjunctive fragment* [13]. We defer details to Sect. 5.

3 Singleton-Preserving Predicate Liftings

Our generic expression language will depend on a specific type of predicate liftings, as well as on a strengthening of separation:

Definition 3.1. An n-ary predicate lifting λ *preserves singletons* if

$$|\lambda_X(\{x_1\},\dots,\{x_n\})| = 1$$

for all $x_1,\dots,x_n \in X$. Moreover, a set Λ of predicate liftings is *strongly expressive* if for every $t \in TX$ there exist $\lambda/n \in \Lambda$ and $x_1,\dots,x_n \in X$ such that

$$\{t\} = \lambda_X(\{x_1\},\dots,\{x_n\}).$$

Singleton preservation will serve to ensure that expressions of our language denote unique behaviours, while strong expressivity will guarantee that all (finite) behaviours are expressible. The following is immediate:

Lemma 3.2. *Every strongly expressive set of predicate liftings is separating.*

Example 3.3. The predicate liftings in Example 2.6 both fail to preserve singletons. Our main source of singleton-preserving predicate liftings are Moss liftings as introduced in general terms in the next section. For the finite powerset functor $\mathcal{P}_\omega$ consider the predicate liftings λ^n/n given by

$$\lambda^n_X(A_1,\dots,A_n) = \{B \in \mathcal{P}_\omega X \mid B \subseteq \textstyle\bigcup_{i=1}^n A_i \text{ and } B \cap A_i \neq \emptyset \text{ for } i = 1,\dots,n\} \tag{3.1}$$

(which can be seen as arising from the above lifting for $\Box$ by Boolean combination). Then $\lambda^n_X(\{x_1\},\dots,\{x_n\}) = \{\{x_1,\dots,x_n\}\}$ for $x_1,\dots,x_n \in X$, which shows that the λ^n preserve singletons and that the set $\{\lambda^n \mid n \in \omega\}$ is strongly expressive.

Remark 3.4. Singleton-preserving predicate liftings should not be confused with Kurz and Leal's *singleton liftings* [18,19]. The definition of the latter is based on the one-to-one correspondence between subsets of $T(2^n)$ and n-ary predicate liftings for T [30], which maps an n-ary predicate lifting λ to $\lambda_{2^n}(\pi_1^{-1}(\{\top\}),\dots,\pi_n^{-1}(\{\top\})) \subseteq T(2^n)$, and $C \subseteq T(2^n)$ to the lifting λ defined by $\lambda_X(A_1,\dots,A_n) = \{t \in TX \mid T\langle\chi_{A_1},\dots,\chi_{A_n}\rangle(t) \in C\}$, where $\pi_i : 2^n \to 2$ is the i-th projection and $\chi_A : X \to 2$ denotes the characteristic function of $A \subseteq X$. An n-ary predicate lifting is a *singleton lifting* if it corresponds to a singleton subset of $T(2^n)$.

It is then indeed immediate that every *unary* singleton-preserving predicate lifting λ is a singleton lifting, since the above correspondence maps λ to the singleton $\lambda_2(\{\top\})$. The following examples show that this implication breaks down at higher arities, and that the converse also fails in general.

Example 3.5. (1) The unary singleton lifting for $\mathcal{P}$ corresponding to $\{\{\bot\}\} \subseteq \mathcal{P}2$ fails to preserve singletons. Of course, this lifting fails to be monotone.

(2) Binary monotone singleton liftings need not preserve singletons. E.g. for the distribution functor $\mathcal{D}$, the monotone singleton lifting λ corresponding to $\{1 \cdot (\top, \top)\} \subseteq \mathcal{D}(2^2)$ is given by $\lambda(A, B) = \{\mu \mid \mu(A) = \mu(B) = 1\}$, so $\lambda(\{x\}, \{y\}) = \varnothing$ for $x \neq y$. We leave it as an open question whether unary monotone singleton liftings preserve singletons.

(3) The binary singleton-preserving predicate lifting

$$\lambda(A, B) = \{\mu \mid \mu(A) \geq 1/2, \mu(B) \geq 1/2, \mu(A \cup B) = 1\}$$

for the distribution functor $\mathcal{D}$ (see Example 4.7 for details) is not a singleton lifting, as it corresponds to the following infinite subset of $\mathcal{D}(2^2)$:

$$\{\mu \mid \mu(2 \times \{\top\}) \geq 1/2, \mu(\{\top\} \times 2) \geq 1/2, \mu(2 \times \{\top\} \cup \{\top\} \times 2) = 1\}.$$

It is not hard to see that we can recover operations for a functor from *monotone* singleton preserving predicate liftings; in detail:

Lemma 3.6. *Let* $T : \mathbf{Set} \to \mathbf{Set}$. *Then the following hold.*

1. *For each monotone singleton-preserving predicate lifting* λ/n,

$$\{\tau_{\lambda,X}(x_1, \ldots, x_n)\} := \lambda_X(\{x_1\}, \ldots, \{x_n\}) \tag{3.2}$$

   defines a natural transformation $\tau_\lambda : (-)^n \to T$.
2. *If* Λ *is a strongly expressive set of monotone singleton-preserving predicate liftings, then taking operation symbols* τ_λ *for each* $\lambda \in \Lambda$, *with associated interpretation as per* (3.2), *yields a functor presentation of* T.

Example 3.7. The singleton-preserving predicate liftings λ^n from Example 2.6 induce, according to the above construction, the operations $X^n \to \mathcal{P}_\omega(X)$, $(x_1, \ldots, x_n) \mapsto \{x_1, \ldots, x_n\}$.

The other direction, generating predicate liftings from functor presentations, is more involved, and treated next.

4 Moss Liftings

Marti and Venema [20] introduce *Moss liftings*, predicate liftings that are constructed from functor presentations with the help of a generalized form of the nabla operator, extending an earlier construction for weak-pullback preserving functors by Kurz and Leal [18]. Recall that for a weak-pullback-preserving functor T, Moss' [24] classical nabla operator $\nabla : T\mathcal{Q} \Rightarrow \mathcal{Q}T^{\mathsf{op}}$ is the natural transformation defined by

$$\nabla(\Phi) = \{t \in TX \mid (t, \Phi) \in \overline{T}(\in_X)\}.$$

Here, $\in_X \subseteq X \times \mathcal{Q}X$ is the element-of relation for X, and $\overline{T}$ is the *Barr extension* of T, viz. the functor $\overline{T}$ on the category of sets and relations defined on a relation $R \subseteq X \times Y$ by $\overline{T}R = \{(T\pi_1(r), T\pi_2(r)) \mid r \in TR\}$, where $\pi_1 : R \to X$ and $\pi_2 : R \to Y$ are the projection maps (cf. [24]). Barr [5] (see also Trnková [37]) proved that $\overline{T}$ is a functor if and only if T preserves weak pullbacks.

Further recall that the *converse* of a relation $R \subseteq X \times Y$ is the relation $R^\circ = \{(y, x) \mid x\ R\ y\} \subseteq Y \times X$. We denote the composite of two relations $R \subseteq X \times Y$ and $S \subseteq Y \times Z$ diagrammatically by $R; S \subseteq X \times Z$. Also, for $A \subseteq X$ we denote by $R[A] \subseteq Y$ the relational image $R[A] = \{y \mid \exists x \in A.\, xRy\}$. The construction $T \mapsto \overline{T}$ is generalized and abstracted in the notions of *relation lifting* and, more specifically, *lax extension* of a functor, as recalled next.

Definition 4.1 (Relation lifting, lax extension [20]). A *relation lifting* L for a functor T is an assignment mapping every relation $R \subseteq X \times Y$ to a relation $LR \subseteq TX \times TY$ such that converses are preserved: $L(S^\circ) = (LS)^\circ$. A relation lifting L is a *lax extension* if for all relations $R, R' \subseteq X \times Z$, $S \subseteq Z \times Y$ and functions $f : X \to Z$ (identified with their graph relation) the following hold:

$$R' \subseteq R \Rightarrow LR' \subseteq LR,$$
$$LR; LS \subseteq L(R; S),$$
$$Tf \subseteq Lf.$$

A lax extension L *preserves diagonals* if for all sets X

$$L\Delta_X \subseteq \Delta_{TX}.$$

Proposition 4.2 (Properties of Lax Extensions [20]). *Let L be a lax extension for a functor T. Then for all functions $f : X \to Z$, $g : Y \to Z$ and relations $R \subseteq X \times Z$, $S \subseteq Z \times Y$,*

(i) $\Delta_{TX} \subseteq L\Delta_X$,
(ii) $Tf; LS = L(f; S)$ *and* $LR; (Tg)^\circ = L(R; g^\circ)$,

and if L preserves diagonals, then

(iii) $\Delta_{TX} = L\Delta_X$ *and* $Tf = Lf$,
(iv) $Tf; (Tg)^\circ = L(f; g^\circ)$.

One use of relation liftings is to determine coalgebraic notions of bisimulation:

Definition 4.3 (L**-Bisimulation [20]).** Let L be a relation lifting for a functor $T : \mathbf{Set} \to \mathbf{Set}$, and let (X, ξ), (Y, ζ) be T-coalgebras. A relation $S \subseteq X \times Y$ is an L*-simulation* if for all $x \in X$ and $y \in Y$,

$$x\ S\ y \quad \text{implies} \quad \xi(x)\ LS\ \zeta(y).$$

An L*-bisimulation* is a relation S such that S and S° are L-simulations. Two states are L*-bisimilar* if there exists an L-bisimulation relating them.

Marti and Venema [20, Theorem 11] show that if L is a lax extension that preserves diagonals, then L-bisimilarity coincides with behavioural equivalence.

Assumption 4.4. From now on we fix a finitary endofunctor $T : \mathbf{Set} \to \mathbf{Set}$ having a diagonal-preserving lax extension L and a presentation (Σ, α) of T.

Another key feature of lax extensions is that they induce canonical modalities, generalizing Moss' coalgebraic logic [24]:

Definition 4.5 (Lax Nabla [20]). The *lax nabla* of L is the family of functions

$$\begin{aligned} \nabla^L_X : T\mathcal{Q}X &\to \mathcal{Q}T^{\mathsf{op}}X \\ \Phi &\mapsto \{t \in TX \mid (t, \Phi) \in L(\in_X)\}, \end{aligned}$$

where $\in_X \subseteq X \times \mathcal{Q}X$ is the element-of relation for X.

As shown by Marti and Venema [20], the lax nabla is in fact a natural transformation $\nabla^L : T\mathcal{Q} \Rightarrow \mathcal{Q}T^{\mathsf{op}}$, and coincides with Moss' classical ∇ for L being the Barr extension of T (and T preserving weak pullbacks). In combination with a functor presentation, the lax nabla gives rise to a family of predicate liftings:

Definition 4.6 (Moss Liftings [20]). Every operation symbol $\tau/n \in \Sigma$ yields a predicate lifting λ defined by

$$\lambda = (\mathcal{Q}^n \overset{\tau\mathcal{Q}}{\Longrightarrow} T\mathcal{Q} \overset{\nabla^L}{\Longrightarrow} \mathcal{Q}T^{\mathsf{op}}),$$

that is,

$$\lambda_X(X_1, \dots X_n) = \{t \in TX \mid (t, \tau_{\mathcal{Q}X}(X_1, \dots, X_n)) \in L(\in_X)\}.$$

These predicate liftings are called the *Moss liftings* of T.

Example 4.7. Some standard functor presentations are converted into Moss liftings as follows.

(1) For the deterministic automata functor $TX = 2 \times X^A$ consider the Barr extension $L = \overline{T}$. Then elements of $T\mathcal{Q}X$ are pairs $(b, (Y_a)_{a\in A})$, where each Y_a is a subset of X, and

$$\nabla_X(b, (Y_a)_{a\in A}) = \{(b, (x_a)_{a\in A}) \mid \forall a \in A : x_a \in Y_a\} \qquad \text{for } b = 0, 1.$$

The two Moss liftings $\lambda^0, \lambda^1 : \mathcal{Q}^A \to \mathcal{Q}(2 \times (-)^A)$ corresponding to the two $|A|$-ary operation symbols from the presentation in Example 2.4(1) are thus defined (slightly abusing notation) by

$$\lambda^i((Y_a)_{a\in A}) = \{(i, (x_a)_{a\in A}) \mid \forall a \in A : x_a \in Y_a\} \qquad \text{for } i = 0, 1.$$

(2) As indicated in Example 2.4, the finite powerset functor $\mathcal{P}_\omega$ has operations τ^n/n given by $\tau^n(x_1, \dots, x_n) = \{x_1, \dots, x_n\}$. The Moss lifting λ^n associated to τ^n when using the Barr extension is exactly the one given by (3.1) above.

(3) Recall from Example 2.4 that the operations of the finite distribution functor $\mathcal{D}$ take formal convex combinations. Via the Barr extension, such an operation, determined by coefficients $p_1, \dots, p_n$ such that $\sum p_i = 1$, induces the predicate lifting λ given by $\lambda_X(A_1, \dots, A_n)$ consisting of all $\mu \in \mathcal{D}X$ such that there exists a distribution on $\in_X$ (a subset of $X \times \mathcal{Q}(X)$) whose marginal distributions are μ (on X) and the distribution ν on $\mathcal{Q}(X)$ given by $\nu(\{A_i\}) = p_i$, respectively. In fact, however, this description can be substantially simplified; e.g. one readily checks that in the case $n = 2$, we actually have

$$\lambda(A_1, A_2) = \{\mu \in \mathcal{D}(X) \mid \mu(A_1) \geq p_1, \mu(A_2) \geq p_2, \mu(A_1 \cup A_2) = 1\}.$$

(The generalization to higher arities is via what is nowadays known as the *splitting lemma* [36, Theorem 11].)

(4) For the finitary monotone neighbourhood functor $\mathcal{M}_\omega$ (Example 2.4), we obtain Moss liftings as follows. Marti and Venema [20] define a diagonal-preserving lax extension L for $\mathcal{M}$ (which, then, restricts to $\mathcal{M}_\omega$) by means of nested Egli-Milner liftings. An explicit description of L is

$$LR = \{(\mathfrak{A}, \mathfrak{B}) \in \mathcal{M}X \times \mathcal{M}Y \mid \forall A \in \mathfrak{A}.\, R[A] \in \mathfrak{B}, \forall B \in \mathfrak{B}.\, R^\circ[B] \in \mathfrak{A}\}$$

for $R \subseteq X \times Y$. In particular, for $\mathfrak{A} \in \mathcal{M}X$ and $\Phi \in \mathcal{M}\mathcal{Q}X \subseteq \mathcal{Q}\mathcal{Q}\mathcal{Q}X$, we have

$$\mathfrak{A} \in \nabla^L_X(\Phi) \quad \text{iff} \quad \mathfrak{A}\, L(\in)\, \Phi \quad \text{iff} \quad \forall \beta \in \Phi.\, \bigcup \beta \in \mathfrak{A} \text{ and } \forall A \in \mathfrak{A}.\, \{B \in \mathcal{Q}X \mid B \cap A \neq \emptyset\} \in \Phi.$$

Combining ∇^L with the presentation of $\mathcal{M}_\omega$ (Example 2.4) produces, for each choice of numbers $n \geq 0$ and $k_1, \dots, k_n \geq 0$, a $\sum_{i=1}^n k_i$-ary Moss lifting λ given by

$$\lambda((A_{ij})_{i=1,\dots,n;j=1,\dots,k_i}) = \{\mathfrak{A} \in \mathcal{M}_\omega X \mid \forall i.\, \bigcup_j A_{ij} \in \mathfrak{A} \text{ and } \forall B \in \mathfrak{A}.\, \exists i.\, \forall j.\, B \cap A_{ij} \neq \emptyset\}.$$

Since $\mathcal{M}_\omega$ preserves finite sets and the box modality $\Box$ as described in Example 2.6 is separating, it is clear that the Moss liftings are expressible using $\Box$ and Boolean operators. Concretely, this works as follows. For readability, we denote the predicate lifting interpreting $\Box$ by $\Box$ as well, similarly for the dual modality $\Diamond$, so that $\Diamond_X(A) := \mathcal{M}X \setminus \Box_X(X \setminus A) = \{\mathfrak{A} \in \mathcal{M}X \mid \forall B \in \mathfrak{A}.\, B \cap A \neq \emptyset\}$. Then the Moss lifting λ as described above can be written as

$$\lambda((A_{ij})) = \bigcap_i \Box_X(\bigcup_j A_{ij}) \cap \bigcap_\pi \Diamond_X(\bigcup_i A_{i\pi(i)})$$

where π ranges over all selection functions assigning to each $i \in \{1, \dots, n\}$ an index $\pi(i) \in \{1, \dots, k_i\}$.

Moss liftings are always monotone [20, Proposition 24]. We show that they also preserve singletons:

Proposition 4.8. *Moss liftings preserve singletons. More specifically, let λ be the Moss lifting induced by $\tau/n \in \Sigma$. Then for all $x_1, \dots, x_n \in X$,*

$$\lambda_X(\{x_1\}, \dots, \{x_n\}) = \{\tau_X(x_1, \dots, x_n)\}.$$

Marti and Venema already establish that the Moss liftings are separating [20, Proposition 25]; we show that they are even strongly expressive:

Proposition 4.9. *The set Λ of all Moss liftings of T is strongly expressive.*

Remark 4.10. Incidentally, this also means that for finitary functors the existence of a separating set of monotone predicate liftings is equivalent to the existence of a strongly expressive set of monotone singleton-preserving predicate liftings. The right-to-left implication is trivial; the converse follows from Propositions 4.8 and 4.9, and the fact that for finitary functors the existence of a separating set of monotone predicate liftings is equivalent to the existence of a lax extension [20].

We have thus seen that given a fixed diagonal-preserving lax extension, from every natural transformation $\tau : (-)^n \to T$ we obtain the corresponding Moss lifting λ^τ/n, which is a monotone singleton-preserving predicate lifting. Conversely, every monotone singleton-preserving predicate lifting λ yields a natural transformation $\tau^\lambda : (-)^n \to T$ (Lemma 3.6(1)). From Proposition 4.8, it is immediate that for $\tau : (-)^n \to T$ we have

$$\tau = \tau^{(\lambda^\tau)}.$$

In particular, taking Moss liftings is an injection from functor operations to monotone singleton-preserving predicate liftings. Conversely, however, $\lambda = \lambda^{(\tau^\lambda)}$ need not hold in general – recall that the construction of Moss liftings depends on the choice of a diagonal-preserving lax extension, and a functor may have more than one such extension. We report an example due to Paul Levy:

Example 4.11. Let M be the monoid of non-negative reals. This monoid in fact forms a division semiring in the expected sense (e.g. [39]), i.e. it is a semiring, and its non-zero elements form a multiplicative group. We note that every division semiring is *refinable* in the sense of Gumm and Schröder [15], i.e. n specified row sums $b_1, \dots, b_n$ and k specified column sums $c_1, \dots, c_k$ that induce the same total sum $d = \sum b_i = \sum c_j$ can always be realized by some $n \times k$-matrix (a_{ij}) – in fact, one can just put $a_{ij} = b_i c_j / d$. Now let $b \in (0, 1)$ be a transcendental number, and let $N \subseteq M$ be generated by b in M as a division semiring. Concretely, elements of N have the form $f(b)/g(b)$ where $f(X)$ and $g(X) \neq 0$ are polynomials with non-negative rational coefficients. In particular, $1 - b \notin N$: If we could write $1 - b$ in the prescribed form $f(b)/g(b)$, then by transcendentality of b, $f(X)/g(X) = 1 - X$, in contradiction to the leading coefficients of f and g being positive.

Both M and N are *positive* ($x + y = 0$ implies $x = y = 0$) and refinable, so that the monoid-valued functors $F = M^{(-)}$ and $G = N^{(-)}$ both preserve weak pullbacks [15]. As recalled above, it follows that in both cases, the Barr extension is functorial, in particular is a diagonal-preserving lax extension. Now diagonal-preserving lax extensions are easily seen to be inherited by subfunctors, so that the Barr extension $\overline{F}$ induces a diagonal-preserving lax extension L

of G. This extension differs from the Barr extension $\overline{G}$; we immediately cast the counterexample in the form that interests us here:

Let $X = \{u, v\}$. Representing elements of GX as formal linear combinations, we have a binary functor operation $\tau(x, y) = x + by$ for G. We write λ^1 and λ^2 for the Moss liftings induced from τ via $\overline{G}$ and via L, respectively (by the above, both λ^1 and λ^2 induce τ). Then $u + bv \in \lambda^1(\{u, v\}, \{u\})$ but $u + bv \notin \lambda^2(\{u, v\}, \{u\})$: For the former, we have a unique witnessing element of $F{\in_X}$, namely $(1 - b)(u, \{u, v\}) + b(v, \{u, v\}) + b(u, \{u\})$; but in $G{\in_X}$, there is no witnessing element since $1 - b \notin N$.

Summing up, even for weak-pullback preserving functors, singleton-preserving monotone predicate liftings are not in general uniquely determined by the functor operation they induce. In the above example, both singleton predicate liftings inducing the given functor operation arise as Moss liftings, via different diagonal-preserving lax extensions; we currently do not know whether every singeleton-preserving monotone predicate lifting is a Moss lifting for some diagonal-preserving lax extension.

Remark 4.12. It is fairly easy to see that for monotone singleton-preserving *unary* predicate liftings λ, we do have $\lambda = \lambda^{(\tau^\lambda)}$.

5 Generic Expressions

We proceed to define, given a set of monotone and singleton-preserving predicate liftings for a functor T, syntactic expressions describing the behaviour of states of T-coalgebras. Our main result is a Kleene-type theorem stating that for every state of a T-coalgebra there exists an equivalent expression, and conversely, every expression describes the behaviour of some state of a finite T-coalgebra. As indicated above, our expression language is a small fragment of the coalgebraic μ-calculus, essentially restricted to modalities and greatest fixed points $\nu z.\, \phi$.

Definition 5.1 (Expressions). We fix a set V of *fixed point variables* and a set $\mathcal{L}$ of *modalities* equipped with an arity function $\mathrm{ar} : \mathcal{L} \to \omega$; we write $L/n \in \mathcal{L}$ if $L \in \mathcal{L}$ and $\mathrm{ar}(L) = n$. The set $\mathcal{E}$ of *expressions* $\phi, \ldots$ is then defined by the grammar

$$\phi ::= z \mid \nu z.\, \phi \mid L(\phi_1, \ldots \phi_n) \qquad (z \in \mathsf{V}, L/n \in \mathcal{L}).$$

An expression is *closed* if all its fixed point variables are bound by a fixed point operator. An expression is *guarded* if all its fixed point variables are separated from their binding fixed point operator by at least one modality. We write $\mathcal{E}_0$ for the set of closed and guarded expressions. We have the usual notion of α-*equivalence* of expressions modulo renaming of bound variables. An occurrence of a fixed point operator in an expression is *top-level* if it is not in scope of a modality.

We next define the semantics of expressions, which agrees with their interpretation as formulas in coalgebraic logic. We fix the requisite data:

Assumption 5.2. For the rest of the paper, we fix a set $\mathcal{L}$ of modalities and an assignment of a singleton-preserving monotone n-ary predicate lifting $[\![L]\!]$ for T to each $L/n \in \mathcal{L}$ such that the set $\Lambda := \{[\![L]\!] \mid L \in \mathcal{L}\}$ is strongly expressive.

By the results of the previous section, these assumptions imply that T has a presentation and is thus finitary (Theorem 2.2).

Definition 5.3 (Semantics). Given a T-coalgebra $C = (X, \xi)$ and a valuation $\kappa : \mathsf{V} \to \mathcal{Q}X$, the semantics $[\![\phi]\!]^\kappa_C \subseteq X$ of expressions $\phi \in \mathcal{E}$ is given by

$$\begin{aligned} [\![z]\!]^\kappa_C &= \kappa(z) \\ [\![L(\phi_1, \dots \phi_n)]\!]^\kappa_C &= \xi^{-1}[[\![L]\!]_X([\![\phi_1]\!]^\kappa_C, \dots [\![\phi_n]\!]^\kappa_C)] \\ [\![\nu z.\phi]\!]^\kappa_C &= \nu Y.[\![\phi]\!]^{\kappa[z \mapsto Y]}_C \end{aligned}$$

where as usual, we use ν to denote greatest fixed points of monotone maps. When ϕ is closed, we simply write $[\![\phi]\!]_C$ in lieu of $[\![\phi]\!]^\kappa_C$, and we drop the subscript C whenever C is clear from the context.

Note that since the predicate liftings $[\![L]\!]$ are monotone and ξ^{-1} is a monotone map, the requisite greatest fixed points exist by the Knaster-Tarski fixed point theorem. Moreover, the assumption that the predicate liftings are singleton-preserving will ensure that every expression describes exactly one behavioural equivalence class (see Theorem 5.15).

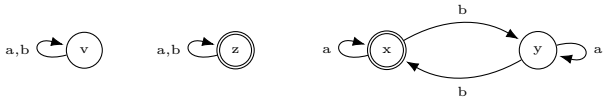
By dint of the fact that our expression language is contained in the coalgebraic μ-calculus, the following is an immediate consequence of the fact that the latter is invariant under behavioural equivalence (e.g. [31]):

Lemma 5.4 (Invariance under behavioural equivalence). *For every closed expression ϕ and coalgebras $C = (X, \xi)$, $D = (Y, \zeta)$, if states $x \in X$ and $y \in Y$ are behaviourally equivalent, then $x \in [\![\phi]\!]_C$ iff $y \in [\![\phi]\!]_D$.*

Lemma 5.5. *For all expressions $\phi \in \mathcal{E}$, $[\![\nu z.\phi]\!] = [\![\phi[\nu z.\phi/z]]\!]$.*

Example 5.6. (1) For the deterministic automaton functor $TX = 2 \times X^A$ with $A = \{a, b\}$, we let $\mathcal{L}$ be the set of two binary modalities $\langle 0, a.(-), b.(-)\rangle$ and $\langle 1, a.(-), b.(-)\rangle$ (corresponding to the two Moss liftings of Example 4.7(1)). We interpret expressions in the final T-coalgebra νT carried by all formal languages over A. Here are a few closed and guarded expressions and their semantics in νT (as usual $|w|_b$ denotes the number of b's in w):

$$\begin{aligned} [\![\nu v.\langle 0, a.v, b.v\rangle]\!] &= \{\emptyset\} \\ [\![\nu z.\langle 1, a.z, b.z\rangle]\!] &= \{A^*\} \\ [\![\nu x.\langle 1, a.x, b.\nu y.\langle 0, a.y, b.x\rangle\rangle]\!] &= \{\{w \in A^* \mid |w|_b \text{ even}\}\} \end{aligned}$$



Note that the semantics of each of these expressions is a singleton (up to behavioural equivalence); in fact, for an arbitrary T-coalgebra X, the semantics of the above expressions is the set of states accepting the language in the singleton on the right. In Lemma 5.12 further below we prove that this holds in general.

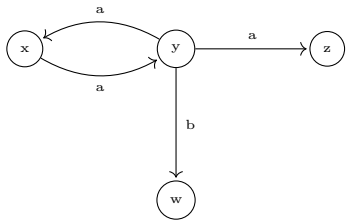
(2) Consider $T = \mathcal{P}_\omega(A \times -)$ where A is a finite set of labels. A presentation of T is given by the signature containing for each n-tuple $\vec{a} = (a_1, \ldots, a_n) \in A^n$ one n-ary operation symbol, and the corresponding natural transformation $\tau^{\vec{a}} : (-)^n \to T$ is defined by

$$\tau^{\vec{a}}_X : (x_1, \ldots, x_n) \mapsto \{(a_1, x_1), \ldots, (a_n, x_n)\}.$$

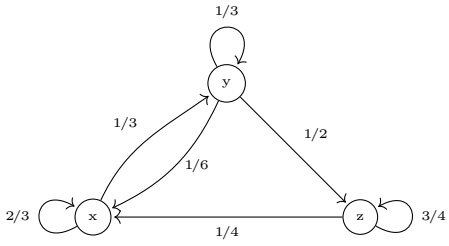
The corresponding Moss lifting is $\lambda^{\vec{a}}/n$ given by

$$\lambda^{\vec{a}}_X(Y_1, \ldots, Y_n) = \{Z \in \mathcal{P}_\omega(A \times X) \mid Z \subseteq \textstyle\bigcup_{i=1}^n (\{a_i\} \times Y_i) \text{ and } Z \cap \{a_i\} \times Y_i \neq \emptyset \text{ for } i = 1, \ldots, n\}$$

(cf. (3.1)). Now put $\mathcal{L} = \{[\vec{a}]/n \mid \vec{a} \in A^n, n \in \omega\}$ and interpret each $[\vec{a}]$ by $\lambda^{\vec{a}}$. For example, for $A = \{a, b\}$ the expression $\nu x.[a]([a, b, a](x, [()], [()]))$, where $[()]$ is the unique nullary modality in $\mathcal{L}$, describes the left-hand state in the following labelled transition system



(3) For $T = \mathcal{D}$ we have the presentation with an n-ary operation $\tau^{\vec{p}}$ for every $\vec{p} = (p_1, \ldots, p_n)$ with $\sum_{i=1}^n p_i = 1$ and corresponding Moss liftings as described in Example 4.7(3). For each such $\vec{p}$, we introduce a modality $[\vec{p}]/n \in \mathcal{L}$, and interpret it as $\lambda^{\vec{p}}$. Now consider the Markov chain (i.e. $\mathcal{D}$-coalgebra)



The behaviour of the left-hand state is described by the expression

$$\nu x.[2/3, 1/3](x, \nu y.[1/6, 1/3, 1/2](x, y, \nu z.[1/4, 3/4](x, z))).$$

Remark 5.7. The syntax of our expressions is determined purely by the finitary coalgebraic type functor, more precisely, by a given strongly expressive set Λ of monotone singleton-preserving predicate lifting. In contrast, existing expression calculi such as standard regular expressions for deterministic automata or the coalgebraic expression calculi in [32,34] use extra operations (e.g. expressing union or concatenation of languages). These operations are not dictated by the setting, viz. an endofunctor on **Set**. Rob Myers' PhD thesis [25] explains nicely how such extra operations are obtained naturally in an expression calculus when one works over an algebraic category (such as the one of join-semilattices or vector spaces over the reals, i.e. algebras for the monad $\mathbb{R}^{(-)}$). We leave the extension of our expression language to this more general setting for future work.

Our Kleene theorem requires a number of technical lemmas:

Lemma 5.8. *Let λ/n and λ'/n' be monotone singleton-preserving predicate liftings for T. Let S be an equivalence relation on a set X, let $A_1, \dots, A_n$, be S-equivalence classes or empty, and let $B_1, \dots, B_{n'}$ be S-closed subsets of X. Then the following holds.*

(1) $\lambda_X(A_1, \dots A_n) \subseteq \lambda'_X(B_1, \dots B_{n'})$ *or* $\lambda_X(A_1, \dots A_n) \cap \lambda'_X(B_1, \dots B_{n'}) = \varnothing$.
(2) If the $B_1, \dots, B_{n'}$ *are even S-equivalence classes or empty, then*

$$\lambda_X(A_1, \dots A_n) = \lambda'_X(B_1, \dots B_{n'}) \text{ or } \lambda_X(A_1, \dots A_n) \cap \lambda'_X(B_1, \dots B_{n'}) = \varnothing.$$

Proof (Sketch). Apply naturality to the quotient map $q : X \twoheadrightarrow X/S$. □

In the proof of Lemma 5.12 further below, we will make use of a Λ-bisimulation. We briefly recall the essentials of this notion [12]:

Definition 5.9 (Λ-Simulation). Given a pair of T-coalgebras (X, ξ) and (Y, ζ), a *Λ-simulation* is a relation $S \subseteq X \times Y$ such that for all predicate liftings $\lambda \in \Lambda$ and $X_i \subseteq X$, $x\, S\, y$ implies

$$\xi(x) \in \lambda_X(X_1, \dots, X_n) \Rightarrow \zeta(y) \in \lambda_Y(S[X_1], \dots, S[X_n]).$$

A *Λ-bisimulation* is a Λ-simulation S such that S° is also a Λ-simulation. Elements $(x, y) \in X \times Y$ are *Λ-bisimilar* if there is a Λ-bisimulation relating x and y.

Theorem 5.10. *Λ-bisimilarity conincides with behavioural equivalence.*

Remark 5.11. In fact, for Theorem 5.10 it is sufficient that Λ is separating and the predicate liftings in Λ are monotone. It turns out that Theorem 5.10 is actually a special case of [20, Theorem 11], applied to the case where the lax extension is induced by a separating set of monotone predicate liftings.

Lemma 5.12. *Let (X, ξ) be a T-coalgebra, let $\lambda_i/k \in \Lambda$, $i = 1, \dots, k$, and let $(A_1, \dots, A_k)$ be the greatest fixed point of the map $h : (\mathcal{Q}X)^k \to (\mathcal{Q}X)^k$ defined by*

$$\begin{pmatrix} X_1 \\ \vdots \\ X_k \end{pmatrix} \mapsto \begin{pmatrix} \xi^{-1}[\lambda_{1,X}(X_1, \dots, X_k)] \\ \vdots \\ \xi^{-1}[\lambda_{k,X}(X_1, \dots, X_k)] \end{pmatrix} \tag{5.1}$$

Then for each i, all elements of A_i are behaviourally equivalent, and for all i, j, either $A_i \cap A_j = \varnothing$ or $A_i = A_j$.

(In the above lemma, we restrict to all λ_i having full arity k and using their arguments in the given order only in the interest of readability; this is w.l.o.g. since we can just reorder arguments and add dummy arguments.)

Proof (Sketch). Let $S \subseteq X \times X$ be the relation

$$S = \{(x_1, x_2) \mid \exists A_i \,.\, x_1 \in A_i \wedge x_2 \in A_i\} \cup \Delta_X.$$

Using Lemma 5.8 one shows first that S is an equivalence relation, which already takes care of the second part of the claim, and then that S is a Λ-bisimulation. The first claim of the lemma then follows by Theorem 5.10. □

The final ingredient of our Kleene-type correspondence is the following adaptation of Bekič's bisection lemma [6]:

Lemma 5.13. *For complete lattices $(X, \leq)$, $(Y, \leq)$ and for every pair of monotone maps $f : X \times Y \to X$ and $g : X \times Y \to Y$, we have*

$$\nu(x, y).(f(x, y), g(x, y)) = (x_0, y_0) \quad \textit{with} \quad \begin{array}{l} x_0 = \nu x.f(x, \nu y.g(x, y)) \\ y_0 = \nu y.g(x_0, y). \end{array}$$

Although in [6] this lemma only covers least fixed points in a slightly different setting, the proof is the same.

Using Lemma 5.13 we can transform every expression $\phi \in \mathcal{E}_0$ into a system of flat equations $(z_1 = \phi_1, \dots, z_k = \phi_k)$ for some $k \in \omega$, i.e. equations without nested modalities or fixed point operators: This is done by first ensuring that every fixed point operator uses a different fixed point variable and then binding every modality that is not nested directly under a fixed point operator with a new fixed point operator using a fresh variable. Thus we can rewrite every expression $\phi \in \mathcal{E}_0$ in the form

$$\phi \equiv \nu z_1.\, L_1(z_1, \nu z_2.L_2(\dots), \dots, \nu z_k.L_k(\dots))$$

for some modalities $L_i \in \mathcal{L}$, $i = 1, \dots, k$. If we now inductively apply Lemma 5.13 and, for readability, additionally normalize every modality to have as many arguments as there are different fixed point variables in such an expression, introducing dummy arguments where necessary, then we can write ϕ as a system

$$\begin{aligned} z_1 &= L_1(z_1, z_2, \dots, z_k) \\ z_2 &= L_2(z_1, z_2, \dots, z_k) \\ &\vdots \\ z_k &= L_n(z_1, z_2, \dots, z_k) \end{aligned} \tag{5.2}$$

of flat equations. Given any coalgebra $C = (X, \xi)$, the above system induces an obvious map of the form (5.1) (replacing z_i by X_i and L_i by $\lambda_i = [\![L_i]\!]$), and the first components of its greatest fixed point is the semantics $[\![\phi]\!]_C$. The following example shows a concrete case.

Example 5.14 (Applying Bekič's bisection lemma). Consider the expression

$$\phi = \nu x.L_1(x, L_2(x), \nu y.L_3(y, \nu z.L_2(z)))$$

In order to transform it as per the procedure indicated, we first need to add a fixed point operator with a fresh variable to the first occurrence of L_2:

$$\phi = \nu x.L_1(x, \nu w.L_2(x), \nu y.L_3(y, \nu z.L_2(z)))$$

Then we can form the equation system for the variables x, w, y, z

$$\begin{aligned} x &= \overline{L}_1(x, w, y, z) = L_1(x, w, y) \\ w &= \overline{L}_2(x, w, y, z) = L_2(x) \\ y &= \overline{L}_3(x, w, y, z) = L_3(y, z) \\ z &= \overline{L}_4(x, w, y, z) = L_2(z) \end{aligned}$$

where we extend $\mathcal{L}$ with additional operators $\overline{L}_i$ having dummy arguments, defined as indicated. The semantics of this equation system in a coalgebra $C = (X, \xi)$ is defined as the greatest fixpoint (A_0, A_1, A_2, A_3) of the map $h : \mathcal{Q}^n X \to \mathcal{Q}^n X$ defined by

$$h : \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{pmatrix} \mapsto \begin{pmatrix} \xi^{-1}[[\![L_1]\!]_X(X_1, X_2, X_3)] \\ \xi^{-1}[[\![L_2]\!]_X(X_1)] \\ \xi^{-1}[[\![L_1]\!]_X(X_2, X_4)] \\ \xi^{-1}[[\![L_2]\!]_X(X_4)] \end{pmatrix}.$$

The semantics of ϕ in C is then $[\![\phi]\!]_C = A_0$.

The following two results together establish a Kleene-type correspondence for the generic expressions of Definition 5.1.

Theorem 5.15. *Every expression $\phi \in \mathcal{E}_0$ describes exactly one behavioural equivalence class, which is moreover realized in a finite coalgebra. Explicitly: there exists a state x in a finite coalgebra such that for every coalgebra C, $[\![\phi]\!]_C$ contains precisely the states of C that are behaviourally equivalent to x.*

Proof. (Sketch). By Lemma 5.4, it suffices to show that any two states (w.l.o.g. in the same coalgebra, using coproducts) satisfying ϕ are bisimilar. Since ϕ can transformed into a system (5.2) of flat equations, this follows by Lemma 5.12. Realization in a finite coalgebra follows from the finite model property of the coalgebraic μ-calculus [8], and alternatively is shown by constructing a model from the variables in a flat equation system. □

Theorem 5.16. *Let $C = (X, \xi)$ be a finite T-coalgebra. For every $x \in X$, there exists an expression $\phi \in \mathcal{E}_0$ such that $x \in [\![\phi]\!]_C$.*

Proof. Let $X = \{x_1, \ldots, x_k\}$ and w.l.o.g. $x = x_1$. Since Λ is strongly expressive, for every $x_i \in X$ there is a modality L_i, w.l.o.g. with arity k and prescribed argument ordering, such that

$$\{\xi(x_i)\} = [\![L_i]\!]_X(\{x_1\}, \ldots, \{x_k\}).$$

That is, the $\{x_i\}$ solve the system $(x_i = L_i(x_1, \ldots, x_k))_{i=1,\ldots,k}$ of flat fixed point equations, so for the greatest fixed point $(A_1, \ldots, A_k)$ of the system, we have $x_i \in A_i$ for every i, in particular $x = x_1 \in A_1$. It now just remains to convert the equation system into an equivalent single expression in the standard manner [7] (incurring exponential blow-up); then $x \in [\![\phi]\!]_C$ as desired. □

Corollary 5.17. *Every expression denotes a behavioural equivalence class of a state in a finite coalgebra, and conversely every such class is denoted by some expression.*

Example 5.18. (1) For the functor $TX = 2 \times X^A$ for $A = \{a, b\}$ consider the coalgebra with carrier $X = \{x_1, x_2\}$ and with coalgebra structure $\xi : X \to 2 \times X^A$ with $\xi(x_0) = (1, (a \mapsto x_0, b \mapsto x_1))$ and $\xi(x_1) = (0, (a \mapsto x_1, b \mapsto x_0))$. Then we clearly have $\{\xi(x_1)\} = \lambda^1(\{x_1\}, \{x_2\})$ and $\{\xi(x_2)\} = \lambda^1(\{x_2\}, \{x_1\})$. Using the syntax of Example 5.6(1) and following the proof of Theorem 5.16, we obtain the following expression for the behavioural equivalence class (i.e. formal language) for x_1:

$$\nu x_1.\langle 1, a.x_1, b.\nu x_2.\langle 0, a.x_2, b.x_1\rangle\rangle.$$

Note that this is the same expression (modulo α-equivalence) as the third expression from Example 5.6(1).

(2) For the functor $\mathcal{P}_\omega(A \times -)$ and $A = \{a, b\}$ the coalgebra $C = (\{x, y, z, w\}, \xi)$ depicted in Example 5.6(2) satisfies the following equations:

$$\{\xi(x)\} = \lambda_C^{(a)}(\{y\}),\ \{\xi(y)\} = \lambda_C^{(a,b,c)}(\{x, w, z\}),\ \{\xi(w)\} = \lambda_C^{()}(),\ \{\xi(w)\} = \lambda_C^{()}()$$

By Theorem 5.16 $\{\{x\}, \{y\}, \{z\}, \{w\}\}$ solves the following system, reusing the same variable names,

$$x = [a](y), \quad y = [a, b, a](x, w, z), \quad w = [()], \quad z = [()]$$

which can be transformed as demonstrated in Example 5.14 to the expression given in Example 5.6(2), describing the behaviour of the state x.

(3) For the functor $T = \mathcal{D}$ consider the expression from Example 5.6(3): $\nu x.[2/3, 1/3](x, \nu y.[1/6, 1/3, 1/2](x, y, \nu z.[1/4, 3/4](x, z)))$, which transforms to the system

$$x = [2/3, 1/3](x, y), \quad y = [1/6, 1/3, 1/2](x, y, z), \quad z = [1/4, 3/4](x, z).$$

By Theorem 5.15 we can construct a coalgebra $C = (\{x, y, z\}, \xi)$ defined by:

$$\{\xi(x)\} = \lambda_X^{(2/3,1/3)}(\{x\}, \{y\})$$
$$\{\xi(y)\} = \lambda_X^{(1/6,1/3,1/2)}(\{x\}, \{y\}, \{z\})$$
$$\{\xi(z)\} = \lambda_X^{(1/4,3/4)}(\{x\}, \{z\})$$

which is exactly the coalgebra depicted in Example 5.6(3) where x is in the behavioural equivalence class of the above expression.

An alternative approach to defining the semantics of expressions is to construct a T-coalgebra structure on the set $\mathcal{E}_0$ of closed and guarded expressions, similarly as in the work of Silva et al. [34] and also Myers [25]. In Theorem 5.21 below we show that this new semantics coincides with the previous one.

Definition 5.19. We define a T-coalgebra $\varepsilon : \mathcal{E}_0 \to T\mathcal{E}_0$ inductively by

$$\varepsilon(L(\phi_1, \ldots, \phi_n)) \in [\![L]\!](\{\phi_1\}, \ldots, \{\phi_n\}) \tag{5.3}$$
$$\varepsilon(\nu x.\phi) = \varepsilon(\phi[\nu x.\phi/x]). \tag{5.4}$$

This is actually a definition of ε because (a) in (5.3), $[\![L]\!]$ preserves singletons and thus there is only one element in $[\![L]\!](\{\phi_1\}, \ldots, \{\phi_n\})$, and (b) for the inductive part (5.4), one can use the number of top-level fixed point operators as a termination measure, which decreases in each step because the fixed points are guarded.

Now recall that a coalgebra $\xi : X \to TX$ is *locally finite* if every $x \in X$ is contained in a finite subcoalgebra of ξ. Locally finite coalgebras are precisely the (directed) unions of finite coalgebras (see [21]). Thus, it follows from Theorem 5.16 that for any $x \in X$ in a locally finite coalgebra $\xi : X \to TX$, there exists a $\phi \in \mathcal{E}_0$ with $x \in [\![\phi]\!]_X$.

Moreover, $\mathcal{E}_0$ is obviously not finite; however, arguing via finiteness of the Fischer-Ladner closure [16] we obtain

Proposition 5.20. *The T-coalgebra $(\mathcal{E}_0, \varepsilon)$ is locally finite.*

The following theorem says that $(\mathcal{E}_0, \varepsilon)$ serves as a canonical model of the expression language:

Theorem 5.21. *For every closed and guarded expression $\phi \in \mathcal{E}_0$ and every state x in a T-coalgebra C, $x \in [\![\phi]\!]_C$ iff x is behaviourally equivalent to ϕ as a state in $(\mathcal{E}_0, \varepsilon)$.*

In particular, the above implies that

$$\phi \in [\![\phi]\!]_{\mathcal{E}_0} \qquad \text{for all } \phi \in \mathcal{E}_0, \tag{5.5}$$

essentially a truth lemma for $\mathcal{E}_0$. For the proof of Theorem 5.21, we note:

Lemma 5.22. *α-Equivalent expressions are behaviourally equivalent as states in $(\mathcal{E}_0, \varepsilon)$.*

*Proof (*Theorem 5.21, *sketch).* It suffices to prove (5.5): The 'if' direction of the claim then follows from invariance of ϕ under behavioural equivalence (Lemma 5.4), and 'only if' is by Theorem 5.15. We generalize (5.5) to expressions ϕ with free variables: Whenever σ is a substitution of the free variables of ϕ and κ a valuation such that $\sigma(v) \in \kappa(v)$ for every free variable v of ϕ, then

$$\phi\sigma \in [\![\phi]\!]^{\kappa}_{\mathcal{E}_0}.$$

We proceed by induction on ϕ, using Lemma 5.22 in the fixpoint case. □

Remark 5.23. To give a concrete example use of the connection between expression languages and modal fixed point logics afforded by the above results, we note that we now obtain an alternative handle on equivalence of expressions that complements the standard approach via partition refinement: Expressions ϕ, ψ are equivalent iff some state described by ϕ (obtained, e.g., via the one of the model constructions in Theorems 5.15 and 5.21) satisfies ψ. Note that the latter is fairly easy to check as long as the modalities are computationally tractable, since ψ otherwise involves only greatest fixed points. This approach is similar to reasoning algorithms in the lightweight description logic $\mathcal{EL}$ [4], where checking validity of $\phi \to \psi$ is reduced to model checking ψ in a minimal model of ϕ; we leave a more detailed analysis to future work.

6 Conclusion and Further Work

We have defined a generic expression language for behaviours of finite set coalgebras based on predicate liftings, specifically on a strongly expressive set of singleton-preserving predicate liftings. There are mutual conversions between such sets of predicate liftings and functor presentations, one direction being via the Moss liftings introduced by Marti and Venema [20]; we have however demonstrated that these fail to be mutually inverse in one direction, i.e. in general not all singleton-preserving predicate liftings are Moss liftings. Our language is presumably equivalent to the set-based instance of Myer's expression language [25]; our alternative presentation is aimed primarily at showing that expression languages embed naturally into the coalgebraic μ-calculus, generalizing well-known results on the relational μ-calculus [2,10,14,35]. The benefit of this insight is to tighten the connection between expression languages and specification logics, e.g. it allows for combining model checking, equivalence checking, and reasoning within a single formalism. On a more technical note, we show, e.g., that one can provide an alternative semantics of expressions by defining a coalgebra structure on expressions, an approach pioneered by Silva et al. [34] and used also by Myers [25]; in the light of the expressions/logic correspondence, this construction is now seen as a canonical model construction for a fragment of the coalgebraic μ-calculus, and the core part of the proof that the two semantics agree becomes just a truth lemma.

An important point for further work is to extend the current setup from the base category **Set** to algebraic categories (such as join semi-lattices or positive convex algebras) in order to generalize our results to expression calculi

involving convenient additional operations (reflecting the ambient algebraic theory) such as addition. A closely related point is the connection with coalgebraic determinization [33]; it should be interesting to see whether our ideas can lead to expression calculi for coarser system equivalences than bisimilarity, such as trace equivalence for transition systems or distribution bisimilarity for Segala systems. Such a generalization might be based on our recent approach to coalgebraic trace semantics via graded monads [22].

References

1. Aceto, L., Hennessy, M.: Termination, deadlock, and divergence. J. ACM **39**, 147–187 (1992)
2. Aceto, L., Ingólfsdóttir, A., Larsen, K., Srba, J.: Reactive Systems: Modelling, Specification and Verification. Cambridge University Press, New York (2007)
3. Adámek, J., Trnková, V.: Automata and Algebras in Categories, Mathematics and Its Applications, vol. 37. Kluwer, Dordrecht (1990)
4. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: International Joint Conference on Artificial Intelligence, IJCAI 2005. Morgan-Kaufmann (2005)
5. Barr, M.: Relational algebras. In: Reports of the Midwest Category Seminar. LNM, vol. 137. Springer (1970)
6. Bekič, H.: Definable operations in general algebras, and the theory of automata and flowcharts. In: Jones, C.B. (ed.) Programming Languages and Their Definition. LNCS, vol. 177, pp. 30–55. Springer, Heidelberg (1984). https://doi.org/10.1007/BFb0048939
7. Bradfield, J., Stirling, C.: Modal logics and mu-calculi. In: Handbook of Process Algebra, pp. 293–332. Elsevier (2001)
8. Cîrstea, C., Kupke, C., Pattinson, D.: EXPTIME tableaux for the coalgebraic mu-calculus. Log. Methods Comput. Sci. **7**(3:3), 33 (2011)
9. Cîrstea, C., Kurz, A., Pattinson, D., Schröder, L., Venema, Y.: Modal logics are coalgebraic. Comput. J. **54**, 31–41 (2011)
10. Godskesen, J., Ingólfsdóttir, A., Zeeberg, M.: Fra Hennessy-Milner logik til CCS-processer. Master's thesis, Aalborg University (1987)
11. Goncharov, Sergey, Milius, Stefan, Silva, Alexandra: Towards a coalgebraic Chomsky hierarchy. In: Diaz, Josep, Lanese, Ivan, Sangiorgi, Davide (eds.) TCS 2014. LNCS, vol. 8705, pp. 265–280. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44602-7_21
12. Gorín, Daniel, Schröder, Lutz: Simulations and bisimulations for coalgebraic modal logics. In: Heckel, Reiko, Milius, Stefan (eds.) CALCO 2013. LNCS, vol. 8089, pp. 253–266. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40206-7_19
13. Gorín, D., Schröder, L.: Subsumption checking in conjunctive coalgebraic fixpoint logics. In: Advances in Modal Logic, AiML 2014. pp. 254–273. College Publications (2014)
14. Graf, S., Sifakis, J.: A modal characterization of observational congruence on finite terms of CCS. Inf. Control **68**, 125–145 (1986)
15. Gumm, H.P., Schröder, T.: Monoid-labeled transition systems. In: Coalgebraic Methods in Computer Science, CMCS 2001. ENTCS, vol. 44, pp. 185–204. Elsevier (2001)

16. Kozen, D.: Results on the propositional μ-calculus. Theor. Comput. Sci. **27**, 333–354 (1983)
17. Kozen, D.: Kleene algebra with tests. ACM Trans. Program. Lang. Syst. **19**, 427–443 (1997)
18. Kurz, A., Leal, R.: Equational coalgebraic logic. In: Mathematical Foundations of Programming Semantics, MFPS 2009. ENTCS, vol. 249, pp. 333–356. Elsevier (2009)
19. Leal, R.: Predicate liftings versus nabla modalities. In: Coalgebraic Methods in Computer Science, CMCS 2008. ENTCS, vol. 203, pp. 195–220. Elsevier (2008)
20. Marti, J., Venema, Y.: Lax extensions of coalgebra functors and their logic. J. Comput. Syst. Sci. **81**(5), 880–900 (2015)
21. Milius, S.: A sound and complete calculus for finite stream circuits. In: Proceedings of the 25th Annual Symposium on Logic in Computer Science (LICS 2010). pp. 449–458. IEEE Computer Society (2010)
22. Milius, S., Pattinson, D., Schröder, L.: Generic trace semantics and graded monads. In: Coalgebraic and Algebraic Methods in Computer Science, CALCO 2015. LIPIcs, vol. 35, pp. 253–269 (2015)
23. Moggi, E.: Notions of computation and monads. Inf. Comput. **93**(1), 55–92 (1991)
24. Moss, L.: Coalgebraic logic. Ann. Pure Appl. Log. **96**, 277–317 (1999)
25. Myers, R.: Rational coalgebraic machines in varieties: languages, completeness and automatic proofs. Ph.D. thesis, Imperial College London (2013)
26. Parikh, R.: Propositional game logic. In: Foundations of Computer Science, FOCS 1983. IEEE (1983)
27. Pattinson, D.: Coalgebraic modal logic: soundness, completeness and decidability of local consequence. Theor. Comput. Sci. **309**, 177–193 (2003)
28. Pattinson, D.: Expressive logics for coalgebras via terminal sequence induction. Notre Dame J. Formal Log. **45**, 19–33 (2004)
29. Peleg, D.: Concurrent dynamic logic. J. ACM **34**, 450–479 (1987)
30. Schröder, L.: Expressivity of coalgebraic modal logic: the limits and beyond. Theor. Comput. Sci. **390**, 230–247 (2008)
31. Schröder, L., Venema, Y.: Completeness of flat coalgebraic fixpoint logics. ACM Trans. Comput. Log. **19**, 4:1–4:34 (2018)
32. Silva, A., Bonchi, F., Bonsangue, M., Rutten, J.: Quantitative Kleene coalgebras. Inf. Comput. **209**, 822–849 (2011)
33. Silva, A., Bonchi, F., Bonsangue, M., Rutten, J.: Generalizing determinization from automata to coalgebras. Log. Methods Comput. Sci. **9**(1:9), 27 (2013)
34. Silva, A., Bonsangue, M., Rutten, J.: Non-deterministic Kleene coalgebras. Log. Methods Comput. Sci. **6**(3:23), 39 (2010)
35. Steffen, B., Ingólfsdóttir, A.: Characteristic formulae for processes with divergence. Inf. Comput. **110**, 149–163 (1994)
36. Strassen, V.: The existence of probability measures with given marginals. Ann. Math. Stat. **36**, 423–439 (1965)
37. Trnková, V.: General theory of relational automata. Fund. Inform. **3**, 189–234 (1980)
38. Venema, Y.: Automata and fixed point logic: a coalgebraic perspective. Inf. Comput. **204**, 637–678 (2006)
39. Weinert, H.J.: On 0-simple semirings, semigroup semirings, and two kinds of division semirings. Semigroup Forum **28**, 313–333 (1984)