UNIVERSAL COALGEBRAS AND THEIR LOGICS

H. PETER GUMM

Abstract. We survey coalgebras as models of state based systems together with their global and local logics. We convey some useful intuition regarding *Set*-functors which leads naturally to coalgebraic modal logic where modalities are validity patterns for the successor object of a state.

1. Introduction

State based systems are natural models in numerous fields of computer science. Moore- and Mealy-Automata are fundamental for the design of hardware systems, acceptors are needed to define and implement the lexical part of programming languages, modifiable objects in object oriented calculi can be seen as systems whose state is being changed by method calls[14]. Nondeterministic Kripke structures provide universal formalisms for modeling distributed systems and protocols, and probabilistic systems are used to model performance issues.

A common feature of all those systems is the presence of some internal state, and of methods for evolving from a given state to a combination of new states. States are usually not directly observable, but only via some observer functions, which reveal certain features of the current state or some other output depending on it.

Strikingly similar notions, questions and results in all of these domains are apparent, but the similarities could for a long time only be discussed on an informal level. Only with the invention and application of Universal Coalgebra [1, 21], has a common framework been created, which, in a mathematically pleasing way, has all mentioned systems as particular instances. With this in mind, the "right" definitions and the pertinent theorems are just facets of a more general mathematical notion, that of a coalgebra. As a side effect, the general viewpoint allows to integrate further examples, such as e.g. topological spaces, whose kinship with state based systems had hitherto not been realized.

Universal coalgebra is dual to universal algebra over the dual of the category of sets. As the category *Set* is not self dual, universal algebra can not simply be translated to deliver a corresponding theory of coalgebras. It can, however, serve as a formidable source of inspiration.

Parallel to the development of coalgebraic structure theory there has always been the question, what "the" appropriate logic for universal coalgebras should be like. Just as inductive datatypes are equipped with an induction principle, terminal coalgebras can be seen as coinductive types, providing a coinduction principle as a proof rule for showing equality of elements. Cofree coalgebras are generalizations of terminal coalgebras, their elements can be interpreted as coequations, and a dual of Birkhoff's theorem can be obtained in the realm of universal coalgebra. However, unlike the situation in universal algebra, coequations do not seem to play such a prominent role in classical applications of universal coalgebra.

Many systems in Computer Science are modeled with Kripke structures, and their desired properties are formulated in various modal languages, such as linear temporal logic (LTL), computation tree logic (CTL) or Hennessy-Milner logic (HML), to name a few. The famous Hennessy-Milner result for image finite Kripke structures establishes a completeness theorem for image finite Kripke-structures [13]. With the work of D. Pattinson and L. Schröder a corresponding modal logic together with a corresponding completeness result is now available for general (finitary) coalgebras.

In this survey, we give an exposition of universal coalgebra with this logical completeness result in mind. The article is self-contained and geared towards Pattinson's and Schröder's completeness result. We do provide complete proofs, but we mostly want to relay an intuitive understanding of the role of type functors, and of modalities, which, we believe, simplifies substantially the original approach by Pattinson [19] and Schröder [22]. A slower paced textbook introduction to much of this material, excluding coalgebraic modal logic, can be found in [15].

2. Coalgebras: Definitions, Examples and fundamental notions

Let $\mathcal{S}et$ be a category of sets and mappings and $F : \mathcal{S}et \to \mathcal{S}et$ an endofunctor. A *coalgebra of type* F, or F*-coalgebra*, is simply a morphism $\alpha : A \to F(A)$. It is common to denote a coalgebra as a pair $\mathcal{A} = (A, \alpha_{\mathcal{A}})$ where $\alpha_{\mathcal{A}} : A \to F(A)$. We shall call A the *universe of* $\mathcal{A}$ and $\alpha_{\mathcal{A}}$ its *structure*. Whenever possible, we shall draw structure morphisms as downward pointing arrows:

$$\begin{tikzcd}A \arrow[d, "\alpha_{\mathcal{A}}"'] \\ F(A)\end{tikzcd}$$

Given two F-coalgebras $\mathcal{A} = (A, \alpha_{\mathcal{A}})$ and $\mathcal{B} = (B, \alpha_{\mathcal{B}})$, a *homomorphism* from $\mathcal{A}$ to $\mathcal{B}$ is a map $\varphi : A \to B$ with $\alpha_{\mathcal{B}} \circ \varphi = F(\varphi) \circ \alpha_{\mathcal{A}}$, i.e. which makes the obvious diagram commutative.

$$\begin{tikzcd}A \arrow[r, "\varphi"] \arrow[d, "\alpha_{\mathcal{A}}"'] & B \arrow[d, "\alpha_{\mathcal{B}}"] \\ F(A) \arrow[r, "F(\varphi)"] & F(B)\end{tikzcd}$$

To check that the class of all $F-$coalgebras with homomorphisms as defined above forms a category requires no more and no less than applying the defining properties of an endofunctor F. We shall call this the category of all F-coalgebras and denote it by $\mathcal{S}et_F$. [1]

Forgetting the structure map α, one obtains the "forgetful" functor $U : \mathcal{S}et_F \to \mathcal{S}et$, and it is easy to check that U creates and preserves all colimits. For instance,

[1]The term *coalgebra* derives from the fact that universal algebras (see[4]) are given by structure maps in the opposite direction. A group, for instance, is a set with a binary operation of multiplication $\cdot$, a unary inverse operation $^{-1}$ and a constant e, satisfying certain equations. These operations can be combined into a single map $G \times G + G + 1 \to G$, and conversely, every such map can be decomposed into three individual operations, so a group is a certain map $F(G) \to G$, where F is the obvious functor with $F(X) = X \times X + X + 1$. Homomorphisms of universal algebras are defined by the same diagram as above, when the structure maps are reversed.

the sum of a collection $(\mathcal{A}_i)_{i\in I}$ of coalgebras is formed on the sum $S = \Sigma_{i\in I}A_i$ in $\mathcal{S}et$. Let $e_i : A_i \to \Sigma_{i\in I}A_i$ be the canonical injections, then the maps $Fe_i \circ \alpha_i : A_i \to F(A_i)$ make $F(\Sigma_{i\in I}A_i)$ a competitor to the sum, which is mediated by the a unique map $\alpha : \Sigma_{i\in I}A_i \to F(\Sigma_{i\in I}A_i)$. It is easy to check that $(\Sigma_{i\in I}A_i, \alpha)$ is indeed the sum of the coalgebras $\mathcal{A}_i = (A_i, \alpha_i)$.

$$\begin{tikzcd}
A_i \arrow[r, "e_i"] \arrow[d, "\alpha_i"'] & \Sigma_{i\in I}A_i \arrow[d, dashed, "\alpha"] \\
F(A_i) \arrow[r, "Fe_i"] & F(\Sigma_{i\in I}A_i)
\end{tikzcd}$$

Likewise, the coequalizer ψ of two coalgebra homomorphisms φ_1 and φ_2 is constructed with its universe being the coequalizer in $\mathcal{S}et$ of φ_1 and φ_2, and its structure map the mediating map to the competitor $F\psi \circ \alpha_{\mathcal{B}}$ of ψ :

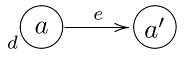
$$\begin{tikzcd}
A \arrow[r, shift left, "\varphi_1"] \arrow[r, shift right, "\varphi_2"'] \arrow[d, "\alpha_{\mathcal{A}}"'] & B \arrow[r, "\psi"] \arrow[d, "\alpha_{\mathcal{B}}"] & C \arrow[d, dashed] \\
F(A) \arrow[r, shift left, "F\varphi_1"] \arrow[r, shift right, "F\varphi_2"'] & F(B) \arrow[r, "F\psi"] & F(C)
\end{tikzcd}$$

With sums and coequalizers existing in $\mathcal{S}et_F$, it follows that all colimits exist in $\mathcal{S}et_F$, see [2].

Most of our structure theoretic statements could be rephrased and proved with $\mathcal{S}et$ replaced by any category $\mathfrak{S}$ satisfying the following assumptions: $\mathfrak{S}$ is complete and co-complete, well-powered and all morphisms are uniquely epi-regular mono decomposable.

2.1. **Automata, objects and transition systems.** Let D be a set of output data and E a set of inputs. In general, an automaton $\mathcal{A} = (A, \delta, \gamma)$ is given by a state transition map $\delta : A \times E \to A$ which takes a state $a \in A$ and an input $e \in E$ to move to a new state $\delta(a, e) \in A$ and some output map γ of varying formats, as follows.

Moore automata. In the case of Moore automata, the output is directly generated from the state using an output map $\gamma : A \to D$. In graphical notation, one depicts the states as nodes of a graph. An arrow labeled by input e connects state a to state a' precisely if $\delta(a, e) = a'$. An output d is written next to state a when $\gamma(a) = d$.



A homomorphism $\varphi : \mathcal{A} \to \mathcal{B}$ between Moore automata $\mathcal{A} = (A, \delta_{\mathcal{A}}, \gamma_{\mathcal{A}})$ and $\mathcal{B} = (B, \delta_{\mathcal{B}}, \gamma_{\mathcal{B}})$ is a map $\varphi : A \to B$ satisfying for all $e \in E$ and for all $a \in A$:

$$\varphi(\delta_{\mathcal{A}}(a, e)) = \delta_{\mathcal{B}}(\varphi(a), e). \tag{2.1}$$

$$\gamma_{\mathcal{A}}(a) = \gamma_{\mathcal{B}}(\varphi(a)) \tag{2.2}$$

Moore-automata are coalgebras for the functor $F(X) = D \times X^E$ which sends a map $f : X \to Y$ to $Ff : D \times X^E \to D \times Y^E$ with $(Ff)(d, \mu) = (d, f \circ \mu)$. Every automaton $\mathcal{A} = (A, \delta, \gamma)$ is an F-coalgebra by virtue of $\alpha(a) = (\gamma(a), \delta(a, -))$, and conversely, from a coalgebra structure $\alpha : A \to D \times A^E$ one recovers the automaton $\mathcal{A} = (A, \delta, \gamma)$ as $\gamma(a) := \pi_1(\alpha(a))$ and $\delta(a, e) := \pi_2(\alpha(a))(e)$. One easily checks that the coalgebraic definition of homomorphism agrees with the standard definition given above.

Acceptors. *Acceptors* are automata with a set $Q \subseteq A$ of *accepting states*. They can be considered as Moore automata with output set $D = \{0, 1\}$ where the output function is $\gamma = \chi_Q$, the characteristic function of the accepting states. Rather than labeling states with either 0 or 1, the accepting states are circled with a double line:

$$\begin{tikzcd} a \arrow[r, "e"] & a' \end{tikzcd}$$

Obviously, acceptors are coalgebras for the functor $F(X) = 2 \times X^E$

Mealy automata. Mealy-Automata take the input into account when generating an output, that is $\gamma : A \times E \to D$. Here inputs as well as outputs are attached to the arrows as, for instance, in the following picture, which encodes $\delta(a, e) = a'$ and $\gamma(a, e) = d$:

$$\begin{tikzcd} a \arrow[r, "e/d"] & a' \end{tikzcd}$$

Mealy-automata are likewise modeled as coalgebras of the functor $F(X) = (D \times X)^E$. The exponent law $(D \times X)^E \cong D^E \times X^E$ emphasizes that Mealy automata could be treated as Moore automata with output set D^E.

Partial automata. Partial automata are permitted to have a partial transition map δ. They are coalgebras as well, but of type $F(X) = D \times (X + 1)^E$ where $1 = \{\star\}$ denotes any one-element set, and "+" is disjoint sum. $\delta(a, e)$ is undefined (denoted by $\perp$) if $\pi_2(\alpha(a))(e) = \star$. One checks that the coalgebraic homomorphism condition amounts to

$$\delta_{\mathcal{A}}(a, e) = \perp \iff \delta_{\mathcal{B}}(\varphi(a), e) = \perp \tag{2.3}$$

in addition to the equations 2.1 and 2.2.

2.2. ***Classes and methods.*** *Classes* in object oriented languages are automata in disguise. State is encapsulated in objects as combinations of values stored in private fields which are not visible to the outside. Output- and transition-functions are realized by *"getter"* and *"setter"* methods, as shown in the following fragment of a Java class modeling rudimentary *Account* objects. Each *Account* can reveal its balance by calling the output-function: *getBalance()* and it can modify its internal state with a method call *deposit(amount)*. We can find out if the state has changed only by calling the method *getBalance()*. The full state may encode further information, such as e.g. the identity of the holder or the bank's billing information. From the outside the *Account* presents just its public methods:

```
class Account{
  int getBalance();
  void deposit(int amount);
}
```

With these methods, observations are restricted to calls to *deposit* followed by *getBalance*. A classical mathematical description, must assume the existence of some internal state set S and must use this as explicit parameter to the methods:

$$\begin{aligned} \mathit{getBalance}: &\quad S &\to \mathbb{Z} \\ \mathit{deposit}: &\quad S \times \mathbb{Z} &\to S \end{aligned}$$

In practice, the state is hidden in the implementation, so it can never be directly observed. Thus a specification such as

$$\mathit{deposit}(\mathit{deposit}(s, z_1), z_2) = \mathit{deposit}(s, z_1 + z_2)$$

is meaningless, since this formula asks for equality of two internal states. The bank may not be willing to guarantee this equality, for instance, if it wants to record transactions for a printout of a monthly statement or if it wants to keep some statistics. It is perfectly acceptable, though, to specify equality of certain observations, as follows:

$$\mathit{getBalance}(\mathit{deposit}(\mathit{deposit}(s, z_1), z_2)) = \mathit{getBalance}(\mathit{deposit}(s, z_1 + z_2)).$$

"Methods" in object oriented languages do not explicitly refer to internal state, so the above specification might be written as follows:

$$\mathit{deposit}(z_1).\mathit{deposit}(z_2).\mathit{getBalance}() = \mathit{deposit}(z_1 + z_2).\mathit{getBalance}().$$

2.3. **Non-deterministic systems.** A *Kripke Structure* is a non-deterministic system $\mathcal{A} = (A, \twoheadrightarrow, l)$ given by a transition relation $\twoheadrightarrow \subseteq A \times A$ and a fixed set Φ of elementary properties. A labeling function $l : A \to \mathbb{P}(\Phi)$ assigns to each state the set of all elementary formulas true in that state. Instead of $(a, b) \in \twoheadrightarrow$, we write $a \twoheadrightarrow b$.

Kripke structures are important devices for modeling hardware, programs, and protocols, as well as distributed systems and games, etc. When the state of a program is given by the value of certain variables, say *x, y, z, PC*, an instruction such as *x:=x+5* modifies the state by incrementing the value of x by 5 and (implicitly) the program counter PC by 1. Properties in Φ are then Boolean terms involving these variables, such as e.g. *"PC < 10"* or *"x*x+y*y < z"*.

Kripke structures can be modeled as coalgebras for the functor $F(X) = \mathbb{P}(X) \times \mathbb{P}(\Phi)$. Here, $\mathbb{P}$ denotes the covariant powerset functor which sends a map $f : X \to Y$ into the *image map* $\mathbb{P}f : \mathbb{P}X \to \mathbb{P}Y$ given by $(\mathbb{P}f)(U) = \{f(u) \mid u \in U\}$. We shall often abbreviate $(\mathbb{P}f)(U)$ by $f[U]$.

Conversely, a coalgebra structure $\alpha : A \to \mathbb{P}(A) \times \mathbb{P}(\Phi)$ decomposes into a binary transition relation $\twoheadrightarrow$ in the first component and a labeling map in the second component.

Bounded Nondeterminism. In many applications, nondeterminism is restricted to a choice from finitely many possible next states; in such a case one speaks of *bounded non-determinism.* In general, a Kripke structure is called *image-finite* if for every state $a \in A$ the set of successors $\alpha(a) := \{a' \mid a \twoheadrightarrow a'\}$ is finite. Image-finite Kripke structures, may be modeled as coalgebras for the finite-powerset functor $\mathbb{P}_\omega$ which is the subfunctor of $\mathbb{P}$ associating with a set X the set $\mathbb{P}_\omega(X)$ of all finite subsets of X.

2.4. **Bisimulations.** In the theory of Kripke-Structures, the notion of *bisimulation* has always played a central role for relating states displaying the same behavior. A bisimulation between Kripke structures relates two states that can mutually simulate each other. To be precise, a bisimulation between Kripke Structures $\mathcal{A} = (A, \to_{\mathcal{A}}, l_{\mathcal{A}})$ and $\mathcal{B} = (B, \to_{\mathcal{B}}, l_{\mathcal{B}})$ is defined as a binary relation $R \subseteq A \times B$ such that $a\,R\,b$ implies

$$l_{\mathcal{A}}(a) = l_{\mathcal{B}}(b) \tag{2.4}$$

$$\forall a' \in A. a \to a' \implies \exists b' \in B.\, b \to b' \wedge a'\,R\,b' \tag{2.5}$$

$$\forall b' \in B. b \to b' \implies \exists a' \in A.\, a \to a' \wedge a'\,R\,b' \tag{2.6}$$

$$\begin{tikzcd}
 & a \arrow[r, dotted, no head] \arrow[dl] & R \arrow[r, dotted, no head] & b \arrow[dr] & \\
a' \arrow[rr, dotted, no head] & & R \arrow[rr, dotted, no head] & & b'
\end{tikzcd}$$

Two states s and s' are called *bisimilar*, written $s \sim s'$, if there exists some bisimulation R with $s\,R\,s'$.

Modal logic. Modal logic is a logical system for talking about Kripke Structures from a local perspective. Modal formulas are generated by the syntax

$$\begin{array}{rcl}
\phi & ::= & p \quad \text{for each}\, p \in \Phi \\
 & | & \Box\phi \mid \Diamond\phi \\
 & | & \mathit{true} \mid \phi \wedge \phi \mid \neg\phi.
\end{array}$$

A state s satisfies $p \in \Phi$ if $p \in l(s)$, and s satisfies $\Box\phi$ (resp. $\Diamond\phi$) if each (resp. some) successor of s satisfies ϕ. Formula induction shows immediately, that bisimilar states are logically equivalent. The converse is true only for image-finite Kripke structures, and it is known as the Hennessy-Milner theorem [13]:

Theorem 1. *Two states s and s' of image-finite Kripke structures are bisimilar if and only if they are logically equivalent.*

Modal logic, interpreted over Kripke structures is called *"normal"*, since it satisfies the following tautology:

$$\Box(\phi \to \psi) \to \Box\phi \to \Box\psi$$

2.5. **Labeled transition Systems.** Variations of Kripke structures are *fuzzy systems*, *weighted systems*, or *probabilistic systems*, where the probabilities attached to the transitions emanating from any state s sum to 1.

▷ Let $\mathcal{L}$ be a complete lattice, and $\mathcal{L}^X$ the set of all maps from X to $\mathcal{L}$. This can be made into a functor by translating a map $f : X \to Y$ into a map $\mathcal{L}^f : \mathcal{L}^X \to \mathcal{L}^Y$ given by $\mathcal{L}^f(\sigma)(y) := \bigvee\{\sigma(x) \mid f(x) = y\}$. The elements of $\mathcal{L}^X$ can be considered as $\mathcal{L}$-fuzzy subsets of X. Indeed, with $\mathcal{L} = \{0, 1\}$ we obtain just the covariant powerset functor $\mathbb{P}$, and with $\mathcal{L} = [0,1]$, the unit interval, $\mathcal{L}^{(-)}$−coalgebras are just fuzzy relations.

▷ Choosing elements m from a monoid $\mathcal{M} = (M, +, 0)$ as weights, we can model weighted transition systems as coalgebras for the functor $\mathcal{M}_\omega^{(-)}$ which sends a a set X to the set of all maps $\sigma : X \to M$ whose support $supp(\sigma) = \{x \in X \mid \sigma(x) \neq 0\}$ is finite. The elements of $\sigma \in \mathcal{M}_\omega^X$ can be considered either as finite bags where $\sigma(x)$ specifies the multiplicity of x in "container" σ, or as formal polynomials $m_1x_1 + ... + m_nx_n$ with variables from X and coefficients from M.

▷ Let $\mathbb{D}(X)$ be the set of all finite probability distributions over X then $\mathbb{D}$ becomes a functor and $\mathbb{D}$-coalgebras are probabilistic transition systems. An appropriate modal logic for probabilistic coalgebras has a modality $[p]$ for each $0 \leq p \leq 1$ with the semantics

$$s \models [p]\phi :\iff p = \Sigma\{p_i \mid s' \models \phi,\, s \overset{p_i}{\to} s'\}.$$

2.6. **Hypersystems.** Neighborhood systems $\mathcal{A} = (A, \nu_\mathcal{A})$ associate with each $a \in A$ a collection of subsets $\nu(a) \subseteq \mathbb{P}(A)$. A morphism between two neighborhood systems $\mathcal{A} = (A, \nu_\mathcal{A})$ and $\mathcal{B} = (B, \nu_\mathcal{B})$ is a map $\varphi : A \to B$ such that for each $a \in A$ and for each subset $Y \subseteq B$ we have

$$Y \in \nu_\mathcal{B}(\varphi(a)) \iff \varphi^{-1}(Y) \in \nu_\mathcal{A}(a).$$

General neighborhood structures turn out to be coalgebras for the functor $F(X) = 2^{2^X}$, which is the composition of the contra-variant powerset functor $2^{(-)}$ with itself. Note that the functors $2^{(-)}$ and $\mathbb{P}(-)$ agree on objects, i.e. 2^X and $\mathbb{P}(X)$ both denote the powerset of X, but they differ on maps, in that a map $f : X \to Y$ is sent to $2^f : 2^Y \to 2^X$ given by $2^f(V) = f^{-1}(V)$ for each $V \subseteq Y$.

Topological spaces are special types of neighborhood systems and neighborhood morphisms between them correspond to maps which are both continuous and open. Topological spaces can also be modeled as coalgebras for the filter functor $\mathbb{F}$, where $\mathbb{F}(X)$ denotes the set of all filters on X. $\mathbb{F}$ is a subfunctor of $2^{2^{(-)}}$ and topological spaces are also $\mathbb{F}$-coalgebras.

Modal logic for neighborhood systems also uses the modalities $\Box$ and $\Diamond$. In this case, a state $a \in A$ satisfies $\Box\phi$ provided $[\![\phi]\!] \in \nu(a)$, where $[\![\phi]\!]$ stands for the set of all $a \in A$ satisfying ϕ. The semantics of $\Diamond\phi$ is defined by the formula $\Diamond\phi = \neg\Box\neg\phi$.

3. Set Functors

$\mathcal{S}et$-functors F are the coalgebraic types, so it is important to know the most relevant properties. First of all, we can assume that $F(X) = \emptyset$ implies $X = \emptyset$, for otherwise F would have to be the constant functor. Since monos in $\mathcal{S}et$ are left-invertible, provided their domain is nonempty, every $\mathcal{S}et$ functor F preserves monos with nonempty domain. More surprising is the following classical result by Trnková [23], a short proof of which can be found in [11]:

Lemma 2. *Every $\mathcal{S}et$ functor F preserves finite nonempty intersections. By redefining F on the empty set and on empty mappings, one can obtain a set functor F^+, which preserves all finite intersections.*

Since empty coalgebras are of no relevance we may assume from now on that F has been "normalized" to preserve *all* monos and *all* finite intersections.

3.1. **Weak pullback preservation.** A *pullback* (P, π_A, π_B) is the limit of a diagram of two arrows f and g with common codomain.

$$\begin{tikzcd}
P \arrow[r, dashed, "\pi_X"] \arrow[d, dashed, "\pi_Y"'] & X \arrow[d, "f"] \\
Y \arrow[r, "g"] & Z
\end{tikzcd}$$

In $\mathcal{S}et$, the pullback of f and g is the set $P = \{(x, y) \in X \times Y \mid f(x) = g(y)\}$ with π_X and π_Y the projection morphisms. If f and g are set-inclusions, then their pullback is the same as the intersection: $P \cong X \cap Y$. Therefore, Trnková's theorem allows us to assume that F preserves pullbacks of inclusions.

For general maps f and g this is not possible. Applying F to the above diagram does not render $(F(P), F\pi_X.F\pi_Y)$ as pullback, not even as a weak pullback of Ff and Fg. This is unfortunate, since in the end it is going to prevent bisimilarity, as we shall define it for general coalgebras, to become an equivalence relation. In turn, we cannot hope for a Hennessy-Milner type theorem, characterizing bisimilarity by logical equivalence, since the latter is by its very nature an equivalence relation.

Weak preservation of pullbacks would be a convenient property for the structure theory of coalgebras, too, and luckily it is satisfied for the type functors of the standard examples that we have seen, including automata, probabilistic systems and Kripke structures. Therefore, almost all of the early literature on universal coalgebra required the type functor F to *weakly preserve pullbacks.*

On the other hand, there are important functors which do not weakly preserve pullbacks, but for which important coalgebraic constructions are still well behaved. The double contra-variant power set functor $2^{2^{(-)}}$, yielding hypersystems, for instance, does not weakly preserve pullbacks, yet bisimilarity of hypersystems is an equivalence relation, thus removing a big obstacle on the way towards a Hennessy-Milner style theorem for hyper systems.

This motivated the search for conditions on type functors that are less restrictive than "weak pullback preservation" and for a characterization of their structure theoretical impact. In [11] it was shown that preservation of weak pullbacks could be decomposed into two simpler preservation conditions

Lemma 3. *A $\mathcal{S}et$-functor F weakly preserves pullbacks iff it weakly preserves preimages, and kernel pairs.*

Here, a *kernel pair* is the pullback of two identical arrows and a *preimage* is a pullback along a mono. The key to the mentioned result is a decomposition, as in the following diagram of an arbitrary pullback diagram of two maps f and g into a kernel diagram, two preimage diagram and an intersection. Preservation of the latter is taken care of by Trnková's theorem.

$$\begin{tikzcd}
X \arrow[r, hook] \arrow[rr, bend left, "f"] & X+Y \arrow[r, "{[f,g]}"] & Z \\
\bullet \arrow[u, dashed] \arrow[r, hook, dashed] & Ker \arrow[u, dashed] \arrow[r, dashed] & X+Y \arrow[u, "{[f,g]}"'] \\
P \arrow[u, hook, dashed] \arrow[r, hook, dashed] & \bullet \arrow[u, hook, dashed] \arrow[r, dashed] & Y \arrow[u, hook] \arrow[uu, bend right, "g"']
\end{tikzcd}$$

In [10, 9, 11], the mentioned preservation properties of a $\mathcal{S}et$-functor F are related to structure theoretic properties of the category $\mathcal{S}et_F$. In particular, it is shown that a functor F weakly preserves kernel pairs if and only if for every constant set C, the largest bisimulation on $C \times F(-)$ - coalgebras is always transitive. As an example, the functor $2^{2^{(-)}}$ does weakly preserve kernel pairs, but not preimages, see [12].

3.2. **Some intuition regarding $\mathcal{S}et$-functors.** A good intuition for $\mathcal{S}et$-functors F is achieved by thinking of the elements $u \in F(X)$ as some mathematical *shapes*, such as sets, trees, graphs, maps, or combinations thereof, containing certain "holes" which are filled with elements of X. A map $f : X \to Y$ leads to a *replacement* or *substitution* map $Ff : FX \to FY$ which in any object $u \in FX$ replaces each occurrence of $x \in X$ by the corresponding object $f(x) \in Y$. The resulting object $(Ff)(u)$ might have to be simplified due to the fact that certain holes end up being filled with equal elements.

Imagine, for instance, a shape with five holes, given by a triple and a set as: $(\circ_1, \circ_2, \{\circ_3, \circ_4, \circ_5\})$. With $x_1, x_2, x_3 \in X$, a corresponding element of $F(X)$ might be $u = (x_1, x_3, \{x_1, x_2, x_3\})$. A map $f : X \to Y$ with $f(x_3) = y_1$ and $f(x_i) = y_i$ otherwise, transforms u to $(Ff)(x_1, x_3, \{x_1, x_2, x_3\}) = (y_1, y_1, \{y_1, y_2, y_1\}) = (y_1, y_1, \{y_1, y_2\})$ in $F(Y)$.

If we formally define a *κ-shape* as any element $p \in F(\kappa)$, it is easy to see that any set of 1-shapes gives rise to a subfunctor of F. For every $s \in F(1)$ let F_s be the functor given by all those $u \in F(X)$ which become equal to s when all holes are filled with the same element, 0, that is: $F_s(X) = \{u \in F(X) \mid F!_X(u) = s\}$, where $!_X : X \to 1$. Then F_s is not only a subfunctor of F but also $F = \Sigma_{s \in F(1)} F_s$ is the sum of all functors F_s, each based on a single shape $s \in F(1)$, see [23].

3.3. **Accessibility.** For any functor F and any cardinal κ, we let F_κ be the subfunctor of F which is based on shapes of cardinality at most κ. A formal definition of this intuitive notion is easy:

$$F_\kappa(X) := \{(F\iota_U)(w) \mid w \in F(U), U \subseteq X, |U| \le \kappa\}, \tag{3.1}$$

where $\iota_U : U \hookrightarrow X$ denotes the natural inclusion of $U \subseteq X$. For each cardinal κ this is a subfunctor of F, and so is $F_{<\kappa} := \bigcup_{\tau<\kappa} F_\tau$. We can recover F as $F(X) = \bigcup_{\kappa \in Ord} F_\kappa$.

Definition 4. F is called *κ-accessible*, if $F = F_{<\kappa}$, and *accessible*, if F is κ-accessible for some κ. F is called *finitary*, if it is ω-accessible.

Every automaton functor $D \times (-)^E$ is κ^+-accessible with $\kappa = |E|$. If a functor F is κ-accessible and $\eta : F \to G$ is a surjective natural transformation, then G is κ-accessible, too.

Since every map $\sigma : \kappa \to X$ factors as $\sigma = \iota_U \circ \sigma'$ with $|U| \leq \kappa$, we readily obtain an alternative representation of F_κ as

$$F_\kappa(X) = \{(F\sigma)(w) \mid w \in F(\kappa),\, \sigma : \kappa \to X\}. \tag{3.2}$$

Fixing κ, we can now consider the automaton functor $F(\kappa) \times (-)^\kappa$ with output set $D = F(\kappa)$ and input set $E = \kappa$. The natural transformation

$$\eta^\kappa : F(\kappa) \times (-)^\kappa \to F(-)$$

given by $\eta^\kappa_X(u, \sigma) = (F\sigma)(u)$ has as image precisely the functor F_κ as described in 3.2. We therefore note:

Theorem 5. *A $\mathcal{S}et$-functor F is accessible iff it is the image under a surjective natural transformation of an automaton functor.*

So, very intuitively, accessible functors are those based on a set (rather than a class) of shapes, whereas finitary functors are based on finite shapes.

3.4. **Separability.** We shall later have reasons to consider 0-1-shapes, i.e. elements of $F(2)$. Every element $u \in F(X)$ determines a set of 0-1-shapes given by $(Ff)(u)$ for any $f : X \to 2$. The question is, whether some $u \in F(X)$ is uniquely representable by the set of its 0-1-shapes. In the coalgebraic logic context, we shall have formulas ϕ determining maps $[\![\phi]\!] : A \to 2$. We shall be interested, whether elements u,$v \in F(A)$ can be separated by some logical formula ϕ, in the sense that $F\ [\![\phi]\!](u) \neq F\ [\![\phi]\!](v)$ are different 0-1-shapes. Temporarily replacing 2 by κ, we define:

Definition 6. A functor is *κ-separable*, if for any pair $u, v \in F(X)$ with $u \neq v$ there exists some map $f : X \to \kappa$ so that $(Ff)(u) \neq (Ff)(v)$.

From the discussion above, we will mainly be interested in *2-separability*, and indeed, most functors, we have seen, are 2-separable:

- ▷ The automaton functor $F(X) = D \times X^E$ is 2-separable. Consider $u = (d_1, \sigma_1) \neq (d_2, \sigma_2) = v$. The case $d_1 \neq d_2$ being trivial, assume $\sigma_1 \neq \sigma_2$, i.e. $\sigma_1(e) \neq \sigma_2(e)$ for some $e \in E$. Choose $f : X \to 2$ with $f(\sigma_1(e)) = 1$ and $f(\sigma_2(e)) = 0$, then $(Ff)(u) = (Ff)(d_1, \sigma_1) = (d_1, f \circ \sigma_1) \neq (d_1, f \circ \sigma_2) = (Ff)(v)$.
- ▷ The powerset functor is 2-separable: Given $u, v \in \mathbb{P}(X)$ with $u \neq v$, there is, w.l.o.g. some $a \in u$ with $a \notin v$. Choose $f : X \to 2$ with $f(a) = 1$ and $f(x) = 0$ for $x \neq a$. Then $(\mathbb{P}f)(u) = f[u] \neq f[v] = (\mathbb{P}f)(v)$.
- ▷ To show that the distribution functor $\mathbb{D}$, the fuzzy logic functor $\mathcal{L}^{(-)}$ and the doubly covariant powerset functor $2^{2^{(-)}}$ are 2-separable, is an easy exercise.

On the other hand, it is not hard to come up with a functor that is not 2-separable. Define, for instance, $F(X) := \{\star\} + \{(x_1, x_2, x_3) \in X^3 \mid |\{x_1, x_2, x_3\}| = 3\}$ with the obvious action on maps. Then F is 3-separable, but not 2-separable, for the trivial reason that $F(2) = \{\star\}$.

4. Properties of $\mathcal{S}et$-Coalgebras

In every category with unique epi-mono decomposition of arrows, epis are *orthogonal* to monos in the following sense: Given a commutative square where e is epi and m is mono, there is a unique diagonal d, making the arising triangles commute.

(4.1)
$$\begin{tikzcd} \cdot \arrow[r, two heads, "e"] \arrow[d] & \cdot \arrow[d] \arrow[dl, dashed, "d"'] \\ \cdot \arrow[r, tail, "m"] & \cdot \end{tikzcd}$$

One can, indeed, obtain d by decomposing the two downward arrows into an epi followed by a mono and then filling in the morphism witnessing uniqueness of the decomposition. We shall first show that the epi-mono decomposition in Set of a homomorphism $\varphi : \mathcal{A} \to \mathcal{B}$ induces an epi-mono decomposition in $\mathcal{S}et_F$ as well.

4.1. Epi-Mono-decomposition in $\mathcal{S}et_F$.

Given a homomorphism of F-coalgebras $\varphi : \mathcal{A} \to \mathcal{B}$, we can decompose the map φ as a $\mathcal{S}et$-map into a surjective φ' followed by an embedding ι as $\varphi = \iota \circ \varphi'$. Applying F and filling in the coalgebra maps, we obtain the perimeter of the following figure. Since, $F\iota$ is necessarily mono, φ' is orthogonal to $F\iota$, so we obtain a coalgebra structure δ on $\varphi[A]$, the image of A under φ, as a diagonal fill in. Clearly, δ is unique in turning both φ' and the embedding ι into homomorphisms. Therefore, the coalgebra $\varphi[\mathcal{A}] := (\varphi[A], \delta)$ is called *the image* of $\mathcal{A}$ under φ. In general, a *homomorphic image* of a coalgebra $\mathcal{A}$ is a coalgebra $\mathcal{B}$ for which there is an epimorphism $\varphi : \mathcal{A} \twoheadrightarrow \mathcal{B}$.

$$\begin{tikzcd} A \arrow[r, two heads, "\varphi'"] \arrow[rr, bend left, "\varphi"] \arrow[d, "\alpha_{\mathcal{A}}"'] & \varphi[A] \arrow[r, tail, "\iota"] \arrow[d, dashed, "\delta"] & B \arrow[d, "\alpha_{\mathcal{B}}"] \\ F(A) \arrow[r] \arrow[rr, bend right, "F\varphi"'] & F(\varphi[A]) \arrow[r, tail, "F\iota"] & F(B) \end{tikzcd}$$

Isomorphisms in $\mathcal{S}et_F$ are the same as bijective homomorphisms and *epi* in $\mathcal{S}et_F$ is the same as surjective homomorphism. *Monomorphisms* in $\mathcal{S}et_F$ need not be injective. In fact, [11] contains a characterization of monos in $\mathcal{S}et_F$, and an example of a coalgebra homomorphism which is both mono and epi, but not iso.

4.2. Subcoalgebras and Cogeneration.

By a *subcoalgebra* $\mathcal{U}$ of a coalgebra $\mathcal{A} = (A, \alpha_{\mathcal{A}})$, we understand a coalgebra defined on a subset $U \subseteq A$, for which the inclusion map $\iota : U \hookrightarrow A$ is a homomorphism. We write $\mathcal{U} \leq \mathcal{A}$ if $\mathcal{U}$ is a subcoalgebra of $\mathcal{A}$.

Given any subset U of A, there is at most one way to define a subcoalgebra structure on U. In that case, we use the term subcoalgebra for that subset, as well. The above diagram shows that every arrow $\varphi : \mathcal{A} \to \mathcal{B}$ in $\mathcal{S}et_F$ factors into $\mathcal{A} \twoheadrightarrow \varphi[\mathcal{A}] \hookrightarrow \mathcal{B}$ where $\varphi[\mathcal{A}]$ is both a homomorphic image of $\mathcal{A}$ and a subcoalgebra of $\mathcal{B}$.

The union of a collection of subcoalgebras is a subcoalgebra, again. Therefore, given any subset $M \subseteq A$, there is always a largest subcoalgebra of $\mathcal{A} = (A, \alpha_{\mathcal{A}})$, which is contained in M. It is called the subcoalgebra of $\mathcal{A}$ *co-generated* by M and denoted by $[M]_{\mathcal{A}}$. The following figure shows how $[M]_{\mathcal{A}} = \bigcup\{A_i \mid \mathcal{A}_i \leq \mathcal{A},\, A_i \subseteq M\}$ arises as the image factorization of the sum $\Sigma\mathcal{A}_i$ of all subcoalgebras $\mathcal{A}_i$ of $\mathcal{A}$, whose universe is contained in M.

$$\begin{tikzcd}
\bigcup A_i \arrow[r, hook] \arrow[rr, bend left, "\iota"] & M \arrow[r, hook] & \mathcal{A} \\
\Sigma\mathcal{A}_i \arrow[u, two heads] & \mathcal{A}_i \arrow[l, "e_i"'] \arrow[u, hook] \arrow[ur, hook]
\end{tikzcd}$$

4.3. **Bisimulations and bisimilarity.** It was Aczel and Mendler, in [1] who carried the important concept of bisimulation to the level of abstract coalgebras for an arbitrary $\mathcal{S}et$-functor F. Their observation was, that bisimulations are relations that can be given a coalgebra structure which agrees with the structure on both components, leading to the following definition:

Definition 7. A relation $R \subseteq A \times B$ is a *bisimulation* between F-coalgebras $\mathcal{A} = (A, \alpha_{\mathcal{A}})$ and $\mathcal{B} = (B, \alpha_{\mathcal{B}})$, provided that an F-coalgebra structure can be defined on R so that the canonical projections π_1 and π_2 become homomorphisms. Elements $a \in A$ and $b \in B$ are called *bisimilar*, and we write $a \sim b$, if $a\,R\,b$ for some bisimulation R.

$$\begin{tikzcd}
A \arrow[d, "\alpha_{\mathcal{A}}"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, dashed, "\rho"'] & B \arrow[d, "\alpha_{\mathcal{B}}"] \\
F(A) & F(R) \arrow[l, "F\pi_1"'] \arrow[r, "F\pi_2"] & F(B)
\end{tikzcd}$$

Example 8. It is straightforward to check that for Kripke structures the above definition agrees with the classical definition given by 2.4, 2.5, and 2.6. For automata we obtain a similar condition, which we choose to formulate in the form of a proof rule:

$$\frac{a\,R\,b}{\gamma(a) = \gamma(b) \wedge \forall e \in E.\delta(a,e)\,R\,\delta(b,e)}. \tag{4.2}$$

It is rather easy to prove that a map $f : A \to B$ is a homomorphism if and only if its graph $G(f) = \{(a, f(a)) \mid a \in A\}$ is a bisimulation. The union of a family of bisimulations between $\mathcal{A}$ and $\mathcal{B}$ is again a bisimulation, hence there is a largest bisimulation $\sim_{\mathcal{A},\mathcal{B}}$. This can be described as follows:

Lemma 9. *$a \sim_{\mathcal{A},\mathcal{B}} b$ iff there exists some coalgebra $\mathcal{R}$, homomorphisms $\varphi_1 : \mathcal{R} \to \mathcal{A}$ and $\varphi_2 : \mathcal{R} \to \mathcal{B}$, and an element $r \in R$ so that $a = \varphi_1(r)$ and $b = \varphi_2(r)$.*

The following picture shows the situation. Notice that it also demonstrates that bisimulations are precisely those relations which can be written as the relational composition $G(\varphi_1)^{-1} \circ G(\varphi_2)$ where φ_1 and φ_2 are homomorphism with common domain.

(4.3)
$$\begin{tikzcd} & r \arrow[dl, "\varphi_1"'] \arrow[dr, "\varphi_2"] & \\ a \arrow[rr, dotted, no head, "\sim"] & & b \end{tikzcd}$$

It would be wrong, however, to conclude that in general, relational compositions of bisimulations would result in bisimulations. In fact, this is not even the case, when we restrict attention to bisimulations *on* a single coalgebra. Even though the largest bisimulation on a coalgebra $\mathcal{A}$, called $\sim_{\mathcal{A}}$, is always reflexive and symmetric, it need not be transitive!

The fact that $\sim_{\mathcal{A}}$ is transitive in the standard examples of automata and Kripke structures, and the desire to factor by the largest bisimulation, has initially lead researchers to impose requirements on the type functor F in order to ensure this property. Trying to prove transitivity, the above characterization requires an element u together with appropriate homomorphisms as in the following figure.

$$\begin{tikzcd} & & u \arrow[dl, dotted] \arrow[dr, dotted] & & \\ & r_1 \arrow[dl] \arrow[dr, "\psi_1"'] & & r_2 \arrow[dl, "\psi_2"] \arrow[dr] & \\ a & & b & & c \end{tikzcd}$$

Thus, given homomorphisms ψ_1 and ψ_2 and elements r_1, r_2 with $\psi_1(r_1) = \psi_2(r_2)$, we need a common preimage u. A way to obtain such a preimage would be to take a weak pullback of ψ_1 and ψ_2. Indeed, $\sim_{\mathcal{A}}$ is transitive, whenever the type functor F weakly preserves pullbacks. (It can be shown that weak preservation of kernel pairs suffices [11])

As mentioned before, most of the functors considered above, prominently the automata functors $D \times (-)^E$, the powerset functor $\mathbb{P}$ and the distribution functor $\mathbb{D}$, weakly preserve pullbacks. It is easy to check that this property is preserved when forming compositions, products and sums.

On the other hand, there remain useful functors, not preserving weak pullbacks, such as the doubly contravariant powerset functor $2^{2^{(-)}}$, or the fuzzy set functor $\mathcal{L}^{(-)}$, unless $\mathcal{L}$ is completely distributive, see [10].

4.4. ***Observational equivalence.*** We have seen that two elements a and b are bisimilar, iff they have a common homomorphic preimage r with $\varphi_1(r) = a$ and $\varphi_2(r) = b$. We can form the pushout (Q, η_1, η_2) of φ_1 and φ_2 , and obtain $\eta_1(a) = \eta_2(b)$. Hence, bisimilar elements have a common homomorphic image, too. This suggests to turn the dual of Lemma 9 into the following definition:

Definition 10. Elements $a \in \mathcal{A}$ and $b \in \mathcal{B}$ are called *observationally equivalent* if there exists some coalgebra $\mathcal{C}$ and homomorphisms $\varphi_1 : \mathcal{A} \to \mathcal{C}$ and $\varphi_2 : \mathcal{B} \to \mathcal{C}$ such that $\varphi_1(a) = \varphi_2(b)$. In that case, we write $a \nabla b$.

$$\begin{tikzcd} a \arrow[rr, dotted, no head, "\nabla"] \arrow[dr, "\varphi_1"'] & & br \arrow[dl, "\varphi_2"] \\ & c & \end{tikzcd}$$

Forming the pushout in diagram 4.3, it follows that $\sim \subseteq \nabla$. By the same token, ∇ is transitive, hence an equivalence relation. Therefore ∇ is a better candidate for a logical characterization than $\sim$.

Early papers on universal coalgebra relied on the notion of bisimilarity for describing "same behavior". In the well studied classical cases of automata and Kripke structures, the "back and forth comparison" in the definition of bisimilarity, 2.5 and 2.6, had always been easily applicable. Observational equivalence was introduced by Kurz in [16], and only gradually was it realized that this notion is preferable to bisimilarity. After all, both notions agree in the classical examples (automata, Kripke structures, hyper-systems).

4.5. **Limits and the terminal coalgebra.** Whereas colimits lift from the base category $\mathfrak{S}$ to the category $\mathfrak{S}_F$ of F-coalgebras, this is not true for limits. Even though the equalizers of two morphisms $\varphi_1, \varphi_2 : \mathcal{A} \to \mathcal{B}$ does exist in $\mathcal{S}et_F$, its universe is usually different from $E(\varphi_1, \varphi_2) = \{a \in A \mid \varphi_1(a) = \varphi_2(a)\}$, the equalizer of φ_1 and φ_2 in the category $\mathcal{S}et$. In fact, the equalizer of φ_1 and φ_2 in $\mathcal{S}et_F$ is the largest coalgebra contained in $E(\varphi_1, \varphi_2)$, see [11]. A similar fact is true for *preimages* i.e. pullbacks along monos.

The most elementary limit which is not guaranteed to exist in $\mathcal{S}et_F$ is the *terminal* coalgebra $\mathcal{T}$, i.e. the product over the empty index set. It turns out that the elements of the terminal coalgebra, if that exists, represent all states of all F-coalgebras, up to bisimilarity. This is the reason, why terminal coalgebras have received special attention. More precisely, assume that the terminal object $\mathcal{T}$ exists in $\mathcal{S}et_F$, then [5]:

Lemma 11. *For every F-coalgebra $\mathcal{A}$ and for every $a \in \mathcal{A}$ there is precisely one element t in $\mathcal{T}$, which is bisimilar to a.*

Proof. Let $\tau_{\mathcal{A}} : \mathcal{A} \to \mathcal{T}$ be the unique homomorphism. Since the graph of $\tau_{\mathcal{A}}$ is a bisimulation, we can choose $t = \tau_{\mathcal{A}}(a)$. Suppose that also $a \sim t'$ for some $t' \in T$, then there must be some coalgebra $\mathcal{R}$, an element $r \in R$ and homomorphisms $\varphi_1 : \mathcal{R} \to \mathcal{A}$ and $\varphi_2 : \mathcal{R} \to \mathcal{T}$ with $\varphi_1(r) = a$ and $\varphi_2(r) = t'$. Now $\tau_{\mathcal{A}} \circ \varphi_1$ and φ_2 are both homomorphisms from $\mathcal{R}$ to the terminal coalgebra, whence $t' = \varphi_2(r) = (\tau_{\mathcal{A}} \circ \varphi_1)(r) = \tau_{\mathcal{A}}(a) = t$.

$$\begin{tikzcd}
 & a \arrow[r, mapsto, "\tau_{\mathcal{A}}"] & t \arrow[d, dotted, no head, "\sim"] \\
r \arrow[ru, dashed, mapsto, "\varphi_1"] \arrow[rr, dashed, mapsto, "\varphi_2"] & & t'
\end{tikzcd}$$

□

4.6. **Lambek's lemma.** In searching for the terminal coalgebra for any functor and in deciding whether it exists, the following result, known as Lambek's Lemma [17] is of central importance:

Lemma 12. *The structure morphism of a terminal coalgebra must be an isomorphism.*

Proof. The proof is immediate from the following diagram which shows the terminal coalgebra $\mathcal{T} = (T, \alpha)$ together with the coalgebra $F\mathcal{T} = (F(T), F\alpha)$ obtained from it by applying the functor F.

$$\begin{tikzcd}
T \arrow[r, "\alpha"] \arrow[d, "\alpha"'] & F(T) \arrow[r, "\beta"] \arrow[d, "F\alpha"] & T \arrow[d, "\alpha"] \\
F(T) \arrow[r, "F\alpha"] & F(F(T)) \arrow[r, "F\beta"] & F(T)
\end{tikzcd}$$

Let β be the unique homomorphism from $F\mathcal{T}$ to $\mathcal{T}$, then all squares commute and $\beta \circ \alpha$ is a homomorphism from $\mathcal{T}$ to itself. Since $\mathcal{T}$ is terminal, $\beta \circ \alpha = id_{\mathcal{T}}$. From the right square, $\alpha \circ \beta = F\beta \circ F\alpha = F(\beta \circ \alpha) = Fid_{\mathcal{T}} = id_{F(T)}$, which proves that β is inverse to α. $\square$

So Lambek's lemma tells us that in order to guess the terminal coalgebra, we should first solve the equation $T \cong F(T)$. This is the key to the following examples:

▷ Consider the functor $F(X) = D \times X$. Coalgebras for F are systems where the transition $\alpha : X \to D \times X$ yields in each step an output $d \in D$ and a new state $x \in X$. The terminal coalgebra for this functor must solve the equation $T \cong D \times T$. Thinking of this definition as a type definition in a programming language, the solution $T = D^\omega$, the set of infinite streams of elements of D suggests itself. Indeed, the coalgebra operation $D^\omega \to D \times D^\omega$ is given by splitting a stream into *head* and *tail*: $\alpha(s) := (hd(s), tl(s))$, and this transition is inverse to the algebra operation $cons : D \times D^\omega \to D^\omega$ where $cons(d, s)$ attaches the element d to the front of stream s. To show that $\mathcal{T} = (D^\omega, \langle hd, tl \rangle)$ is indeed terminal, consider any other F-coalgebra $\mathcal{A} = (A, \alpha_{\mathcal{A}})$. We can split $\alpha_{\mathcal{A}} : A \to D \times A$ into the components $\gamma : A \to D$ and $\delta : A \to A$. To check whether there is a unique homomorphism, we have to verify that the following diagram defines a unique map φ :

$$\begin{tikzcd}
 & D & \\
A \arrow[ur, "\gamma"] \arrow[rr, dashed, "\varphi"] \arrow[d, "\delta"'] & & D^\omega \arrow[ul, "hd"'] \arrow[d, "tl"] \\
A \arrow[rr, dashed, "\varphi"] & & D^\omega
\end{tikzcd}$$

Indeed, for every $a \in A$ this amounts to the equations
– $hd(\varphi(a)) = \gamma(a)$
– $tl(\varphi(a)) = \varphi(\delta(a))$

which defines $\varphi(a)$ uniquely as the stream $(\gamma(a), \gamma(\delta(a)), \ldots, \gamma(\delta^n(a)), \ldots)$.

▷ Adding a set of inputs, consider the automaton functor $F(X) = D \times X^E$. The terminal coalgebra requires solving the equation $T \cong D \times T^E$. Again, it is easy to guess as a solution the set of infinite E-branching trees whose nodes are labeled by elements of D. The structure map yields the pair consisting of the root label $d \in D$ and the E-collection of subtrees of the root. Technically, such trees can be represented as $T = D^{E^\star}$ where $E^\star$ is the set of all words over the alphabet E. (The elements of $E^\star$ are freely generated by starting with the empty word ε and by prefixing a new letter $e \in E$ to an already existing word $w \in E^\star$, yielding a new word $e.w \in E^\star$.) Each word $w \in E^\star$ can be interpreted as a path from the root to a unique node in an E-branching tree, representing that node's unique address, hence each element σ of $D^{E^\star}$ is a map $\sigma : E^\star \to D$, labeling each node with some

element from D.
The coalgebra structure on $T = D^{E^\star}$, written as automaton $\mathcal{T} = (D^{E^\star}, \delta_\mathcal{T}, \gamma_\mathcal{T})$, is given by $\gamma_\mathcal{T}(t) := t(\varepsilon)$ and $\delta_\mathcal{T}(t,e)(w) = t(e.w)$ for each $t \in D^{E^\star}$, $e \in E$ and $w \in E^\star$. To see that $\mathcal{T}$ is indeed the terminal automaton, consider an arbitrary automaton $\mathcal{A} = (A, \delta_\mathcal{A}, \gamma_\mathcal{A})$, then a homomorphism $\varphi : \mathcal{A} \to \mathcal{T}$ is required to satisfy
- $\varphi(a)(\varepsilon) = \gamma_\mathcal{T}(a) = \gamma_\mathcal{A}(a)$
- $\varphi(a)(e.w) = \delta_\mathcal{T}(\varphi(a), e)(w) = \varphi(\delta_\mathcal{A}(a,e))(w)$

which again is a recursive definition of $\varphi(a)$.

▷ An interesting special case is obtained by reducing the output set to $D = \{0,1\}$. An E-branching $\{0,1\}$-labeled tree is uniquely described by the set of all nodes whose label is 1. The addresses of these nodes make up a language $L \subseteq E^\star$and conversely, each language $L \subseteq E^\star$ arises from an E-branching tree whose node at address w is labeled 1 iff $w \in L$. Therefore, the universal acceptor is given as $\mathcal{T} = (\mathbb{P}(E^\star), \delta, Q)$ where for $L \subseteq E^\star$:
- $L \in Q \iff \varepsilon \in L$
- $\delta(L,e) = \{w \in E^\star \mid e.w \in L\} =: L_e$.

L_e is also known as the e-derivative of language L.

So Lambek's lemma helps us on the one hand, to approach the terminal F-coalgebra as a solution to the equation, $T \cong F(T)$, but it also can serve to refute the existence of a terminal coalgebra due to set theoretic reasons. In particular, it follows that no terminal Kripke structure can exist, simply because there can be no set T with $T \cong \mathbb{P}(T)$.

Things can be alleviated, if we consider finitely branching transition systems, where each state has only finitely many successors. These transition systems can be considered as coalgebras of type $\mathbb{P}_\omega(-)$, where $\mathbb{P}_\omega(X)$ denotes the set of all finite subset of X.

The size issue is in fact the only problem which inhibits the existence of the terminal coalgebra. It has been shown by Aczel and Mendler that the terminal coalgebra does exist, if we allow its base set to be a proper class, see [1]. Without resorting to class based coalgebras, it is often possible to consider a given family of F-coalgebras as F_κ-coalgebras, for sufficiently large κ. Now F_κ is accessible, and for accessible functors, the terminal coalgebra always does exist, as we shall see below.

4.7. **Terminal coalgebras for accessible functors.** The underlying idea is very plausible from Kripke-Structures. Even if there is no terminal Kripke-structure, there is a terminal object for the subcategory of Kripke structures with out-degree at most κ. Given any *set* of Kripke structures, we can choose κ larger than the out-degree of all structures involved and consider these as $\mathbb{P}_\kappa$- coalgebras, rather than as $\mathbb{P}$-coalgebras.

Recall that accessibility means that there is a surjective natural transformation

$$\eta^\kappa : F(\kappa) \times (-)^\kappa \twoheadrightarrow F(-).$$

Obviously the first functor $G(-) := F(\kappa) \times (-)^\kappa$ can be considered as automaton functor with input set κ and output set $F(\kappa)$. Therefore, it has a terminal coalgebra $\mathcal{T}_G = (T, \alpha_\mathcal{T})$ as seen above. From this information, we shall construct a terminal coalgebra for F.

In general, note that if $\eta : G \twoheadrightarrow F$ is a surjective natural transformation, then thanks to the axiom of choice, every F-coalgebra $\mathcal{A}_F = (A, \alpha)$ can be obtained from some suitable G-coalgebra $\mathcal{A}_G = (A, \beta)$ by extending its structure with η_A, i.e. $\alpha = \eta_A \circ \beta$. This is enough to verify that the structure of any (weakly) terminal G-coalgebra $\mathcal{W}_G = (W, \alpha_{\mathcal{W}})$ can be extended by η_W to yield a weakly terminal F-coalgebra $\mathcal{W}_F = (W, \eta_W \circ \alpha_{\mathcal{W}})$:

$$\begin{tikzcd}
A \arrow[r, "\tau"] \arrow[d, dotted, "\beta"] \arrow[dd, bend right=60, "\alpha"'] & W \arrow[d, "\alpha_{\mathcal{W}}"] \\
G(A) \arrow[r, "G(\tau)"] \arrow[d, "\eta_A"] & G(W) \arrow[d, "\eta_W"] \\
F(A) \arrow[r, "F(\tau)"] & F(W)
\end{tikzcd}$$

The final step is, to turn a weakly terminal coalgebra $\mathcal{W}$ into a terminal one. This, however, is simply a matter of factoring by the largest congruence or, equivalently, by taking the pushout of all homomorphisms leaving $\mathcal{W}$, which is the same as forming $\mathcal{T} := \mathcal{W}/\nabla_{\mathcal{W}}$. Altogether we therefore have:

Theorem 13. *If F is an accessible functor, then there exists a terminal F-coalgebra.*

So, terminal coalgebras exist, if the functor is accessible. Otherwise their universe might be a proper class, which is really a set theoretical technicality. In fact, given a coalgebra $\mathcal{A} = (A, \alpha_{\mathcal{A}})$ and $a \in A$ let the *out-degree* of a be the smallest κ such that $\alpha_{\mathcal{A}}(a) \in F_\kappa(A) \subseteq F(A)$. The out-degree of $\mathcal{A}$ is then the supremum of all out-degrees of all $a \in A$.

As stated before for Kripke structures, given a *set* of F-coalgebras, we can find some bound κ on their out-degrees, and consider them all as F_κ-coalgebras. In this case the terminal F_κ-coalgebra can for most purposes serve as replacement for a terminal F-coalgebra, see [8].

5. Logics

5.1. **Behaviors.**

We are dealing with systems, that may accept some input, causing them to change their state or to generate new states and to create some output. States are usually not observable, but only their output behavior occurring as a response to the inputs. So what exactly is a behavior?

There are good reasons to identify behaviors with the elements of the terminal coalgebra. We have seen, that every element $a \in \mathcal{A}$ of every coalgebra $\mathcal{A}$ is bisimilar to precisely one element in $\mathcal{T}$. The same holds if we replace bisimilarity by observational equivalence.

- ▷ Consider the case of acceptors with input alphabet E. We can feed a sequence of inputs after which we can decide whether we have reached an accepting state or not. So two states s and s' display the same behavior iff no matter whether starting from s or from s', the same words are accepted. Therefore, the behavior of a state s is encoded by the language $L(s)$ which the automaton accepts when starting from initial state s. Indeed, the terminal acceptor is the set of all languages over E and the terminal morphism is the map $s \mapsto L(s)$.

- ▷ In the case of the functor $F(X) = D \times X$, the coalgebra structure yields a (hidden) state $s' = \delta(s)$ and an observable output $d = \gamma(s)$. Continuing, the observable behavior of a state is the infinite series of outputs $(\gamma(s), \gamma(\delta(s)), \gamma(\delta(\delta(s))), ...,) \in D^\omega$. Now D^ω with operations hd and tl is nothing but the terminal coalgebra for the functor $F(X) = D \times X$.
- ▷ Coalgebras for the functor $F(X) = D \times X + 1$ can in each step starting from state s either yield an output $d \in D$ and change into a new state s' or they can choose to stop. Therefore, an observation may yield either a finite sequence $\sigma \in D^\star$ or an infinite sequence $\tau \in D^\omega$, and indeed $D^\infty = D^\omega + D^\star$, is the basis for the terminal coalgebra. The coalgebra structure $\alpha : D^\infty \to D \times D^\infty + 1$ is given by hd and tl as before, with the special case of the empty sequence $[\,] \in D^\star \subseteq D^\infty$ being mapped to $\alpha([\,]) = 0 \in 1$. α is invertible, by Lambek's lemma, and its inverse is given by the obvious maps $cons : D \times D^\infty \to D^\infty$ and $nil : 1 \to D^\infty$.

5.2. **Coinduction.** Whereas properties of initial datatypes are proved by induction, following the construction of all possible data objects, terminal co-datatypes allow for a *principle of co-induction*, which can be phrased as inference rule

$$\frac{t \sim t'}{t = t'}. \tag{5.1}$$

This means that in order to prove two objects of the co-datatype to be the same, it is enough to show that they are bisimilar. And in order to show that they are bisimilar, it is enough to come up with some bisimulation R with tRt'. The role of the relation R is dual to that of the inductive hypothesis in an induction proof. Proving that R is a bisimulation is dual to the inductive step, and the conclusion is that $t_1 = t_2$. So the coinductive principle, which is valid for the elements of any terminal coalgebra is:

In order to show that $t_1 = t_2$:

Hyp:	invent $R \subseteq T^2$ with $t_1 R t_2$
Step:	show that R is a bisimulation
Con:	conclude $R = id$, hence $t_1 = t_2$.

Example 14. "Lazy" functional languages provide the co-datatype "stream". A stream is given as a pattern $p = h : t$, where $h = hd(p)$ and $t = tl(p)$. Infinite streams can be defined co-recursively, such as e.g. in the following interaction:

```
ones = 1 : ones
from n = n : from (n+1)
plus a:as b:bs = (a+b): plus as bs
```

Given this program, which defines a stream "`ones = (1,1,1,...)`", a unary function "*from*" producing the stream of all integers beginning with some n, and an operation "*plus*" that adds two streams componentwise. We would like to prove:

Claim: `plus ones from 0 = from 1`

The *co-inductive hypothesis* generalizes our claim:

$\forall n \in \mathbb{N}$.`plus ones (from n)` $\sim$ `from (n+1)`

that is, we claim the following to be a bisimulation:

Hyp: $R = \{($ `plus ones (from n), from (n+1)` $\mid n \in \mathbb{N}\}$

For the "Step" we must check rule 4.2 with E trivial and $\gamma = hd$, $\delta = tl$:

Step1: `hd(plus ones (from n)) = (1+n) = hd(from (n+1))`
Step2: `tl(plus ones (from n)) = plus ones (from (n+1)) R from (n+2)`
`= tl(from (n+1))`
Con: $\forall n \in \mathbb{N}$`.plus ones (from n) = from (n+1)`

Rutten considers in [20] the example of coinductive proofs of equations between regular expressions:

Example 15. The states of the terminal acceptor are the languages $L \subseteq E^\star$ where L is accepting iff $\varepsilon \in L$ and where $\delta(L, e) = L_e$ is the e-derivative of L. On the set of all languages we have the classical operation of sum: $L+M = L \cup M$, concatenation: $L \cdot M = \{u \cdot v \mid u \in L,\, v \in L\}$, and Kleene-Star: $L^\star = \{u_1 \cdot \ldots \cdot u_n \mid n \in \mathbb{N}, u_{i,} \in L\}$. The trivial languages are: $0 := \emptyset$ and $1 := \{\varepsilon\}$.

It is rather easy to derive the *rules of differentiation*:

$$\begin{aligned} 1_e &= 0 \\ (L+M)_e &= L_e + M_e \\ (L^\star)_e &= (L \cdot L^\star)_e \end{aligned}$$

According to 4.2, a bisimulation is then a relation R satisfying the proof rule

$$\frac{L\,R\,M}{(\varepsilon \in L \iff \varepsilon \in M) \land \forall e \in E. L_e\, R\, M_e}.$$

A coinductive proof for $(1 + L \cdot L^\star) = L^\star$ can choose the relation

$$R := \{(1 + LL^\star, L^\star) \mid L \subseteq \Sigma^\star\} \cup \{(L, L) \mid L \subseteq L^\star\}$$

as coinductive hypothesis. To show that this is indeed a bisimulation, follows straightforwardly from the above rules.

5.3. **Cofree Coalgebras.** Terminal coalgebras encode behaviors and they satisfy a co-inductive principle, just as dually the initial algebra is an inductively defined object. Since, more generally, algebraic theories are specified by equations, one is led to consider a notion of coequation suitable for delivering a result dual to that famous theorem of Birkhoff, which states that a class of algebras is equationally defined if and only if it is a variety, i.e. closed under homomorphic images, subalgebras and products.

Let Z be a set, the elements of which we shall call *colors* or *covariables.* A coalgebra $\mathcal{T}_Z = (T_Z, \alpha_T)$ together with a "coloring" map $\varepsilon_Z : T_Z \to Z$ is called *cofree over* Z, provided that for any F-coalgebra $\mathcal{A}$ with coloring map $g : A \to Z$ there exists precisely one homomorphism $\tilde{g} : \mathcal{A} \to \mathcal{T}_Z$ such that $g = \varepsilon_Z \circ \tilde{g}$.

$$\begin{tikzcd} & Z & \\ \mathcal{A} \arrow[ur, "g"] \arrow[rr, dashed, "\tilde{g}"] & & \mathcal{T}_Z \arrow[ul, "\varepsilon_Z"'] \end{tikzcd}$$

Note that (the map) ε_Z is left cancellative with respect to homomorphisms $\varphi, \varphi' : \mathcal{A} \to \mathcal{T}_Z$, i.e. $\varepsilon_Z \circ \varphi = \varepsilon_Z \circ \varphi'$ implies $\varphi = \varphi'$. It is also immediate to check that every homomorphism $\varphi : \mathcal{U} \to \mathcal{T}_Z$ from a subcoalgebra $\mathcal{U} \leq \mathcal{A}$ can be extended to a homomorphism $\varphi' : \mathcal{A} \to \mathcal{T}_Z$ with $\varphi = \varphi' \circ \subseteq$.

The terminal F-coalgebra is the same as the cofree F-coalgebra over the one-element color set $Z = \{0\}$, and conversely, the cofree F-coalgebra over color set Z is just the terminal coalgebra over the functor $Z \times F(-)$, for Z constant.

5.4. **Covarieties.** Given a class $\mathfrak{K}$ of F-coalgebras, we shall denote by $\mathcal{H}(\mathfrak{K})$, $\mathcal{S}(\mathfrak{K})$ and $\Sigma(\mathfrak{K})$ the class of all homomorphic images, subcoalgebras and all sums of coalgebras in $\mathfrak{K}$. A *covariety* is a class $\mathfrak{K}$ of coalgebras which is closed under taking homomorphic images, sums and subcoalgebras. Given any class $\mathfrak{K}$ of coalgebras, the smallest covariety containing $\mathfrak{K}$ turns out to be $\mathcal{SH}\Sigma(\mathfrak{K})$, and a class $\mathfrak{K}$ is a covariety, if and only if $\mathfrak{K} = \mathcal{SH}\Sigma(K)$, see [6]. Examples of covarieties are the class of all F-coalgebras, the class of all F-coalgebras with out-degree below κ (see [8]), the class of all topological spaces as $\mathbb{F}$-coalgebras [7].

Let $\mathcal{C}$ be a coalgebra with subcoalgebra $\mathcal{U} \leq \mathcal{C}$ and let

$$\mathfrak{K}(\mathcal{C},\mathcal{U}) = \{\mathcal{A} \in \mathcal{S}et_F \mid \forall \varphi : \mathcal{A} \to \mathcal{C}.\exists \varphi' : \mathcal{A} \to \mathcal{U}.\, \varphi = \iota \circ \varphi'\}$$

be the class of all coalgebras $\mathcal{A}$, such that every homomorphism $\varphi : \mathcal{A} \to \mathcal{C}$ factors through the inclusion ι of $\mathcal{U}$ in $\mathcal{C}$. It is immediate that $\mathfrak{K}(\mathcal{C},\mathcal{U})$ is closed under sums. To show closure under homomorphic images, one uses the fact that epis are orthogonal to monos in $\mathcal{S}et_F$ (see diagram 4.1). Closure under subcoalgebras requires $\mathcal{C}$ to be cofree, so

Lemma 16. *If $\mathcal{T}_X$ is cofree and $\mathcal{U} \leq \mathcal{T}_X$, then $\mathfrak{K}(\mathcal{T}_X,\mathcal{U})$ is a covariety.*

5.5. **Coequations.** Elements of terminal coalgebras $\mathcal{T}$ represent behaviors, so elements of cofree coalgebras $\mathcal{T}_X$ are behaviors with respect to the functor $F(-) \times X$. Relative to F, we consider them as *behavior patterns* p, where the elements of X can be considered as *covariables*. It turns out that each covariety is indeed determined by a set of behavior patterns, thus yielding a dual to Birkhoff's famous theorem in Universal Algebra. For this reason we define:

A *coequation* with covariables from X is an element $p \in \mathcal{T}_X$, the cofree coalgebra over X. Let $\mathcal{A}$ be a coalgebra, $a \in \mathcal{A}$ and $g : A \to X$ a coloring. We define a satisfaction relation $\models$ between elements of $\mathcal{A}$ and patterns p by

$$\mathcal{A}, a \models_g p \;:\Longleftrightarrow\; \tilde{g}(a) \neq p.$$

We further put:

$$\begin{array}{lcl} \mathcal{A}, a \models p & \Longleftrightarrow & \forall g : A \to X.\, \mathcal{A}, a \models_g p \\ \mathcal{A} \models p & \Longleftrightarrow & \forall a \in A.\, \mathcal{A}, a \models p \end{array}$$

Notice that we defined satisfaction of a pattern as "avoidance" of that pattern. This choice permits us to give semantics to a single coequation as an element $p \in T_X$. Alternatively, one might have defined a coequation Q as a *subset* of T_X and required that $\tilde{g}(a) \in Q$. We find it more attractive though, to give a semantics to single elements rather than to subsets only. After all, the characterization of classes of structures via "forbidden patterns" follows well established mathematical traditions in graph theory, algebra, topology, and other fields.

Given a set P of coequations, and a class $\mathfrak{K}$ of F-coalgebras. Define

$$\mathcal{M}od(P) := \{\mathcal{A} \in \mathcal{S}et_F \mid \forall p \in P.\mathcal{A} \models p\}$$

$$\mathcal{C}eq_Z(\mathfrak{K}) := \{p \in \mathcal{T}_Z \mid \forall \mathcal{A} \in \mathfrak{K}.\mathcal{A} \models p\}.$$

as the class of all models of P and the set of all coequations with variables in Z satisfied by all members of $\mathfrak{K}$. With these definitions one obtains a perfect dual to Birkhoff's theorem in universal algebra:

Theorem 17. *Let F be κ-accessible, then for every class of coalgebras one has $\mathcal{M}od(\mathcal{C}eq_\kappa(\mathfrak{K})) = \mathcal{SH}\Sigma(\mathfrak{K})$.*

Proof. With the help of Lemma 16, it is rather easy to verify that for every set P of patterns $\mathcal{M}od(P)$ is always a covariety. It follows that $\mathcal{M}od(\mathcal{C}eq_\kappa(\mathfrak{K})) \supseteq \mathcal{SH}\Sigma(\mathfrak{K})$. For the converse, let $\mathcal{A} \in \mathcal{M}od(\mathcal{C}eq_\kappa(\mathfrak{K}))$ be given, then we find for every $a \in A$ a subcoalgebra $\mathcal{U}_a$ with $a \in \mathcal{U}_a$ and $|U_a| < \kappa$. Obviously, $A = \bigcup_{a\in A} U_a$, so there is an epi $\Sigma_{a\in A}\mathcal{U}_a \twoheadrightarrow \mathcal{A}$. As $\mathcal{SH}\Sigma(\mathfrak{K})$ is closed under sums and homomorphic images, it is enough, to show that each $\mathcal{U}_a$ is contained in $\mathcal{SH}\Sigma(\mathfrak{K})$.

Since $|U_a| < \kappa$, there is an injective map $g : U_a \to \kappa$, hence $\tilde{g}[\mathcal{U}_a]$ is a subcoalgebra of $\mathcal{T}_\kappa$ which is isomorphic to $\mathcal{U}_a$. Every element of $p \in \mathcal{U}_a$ is a coequation which is obviously violated by $\mathcal{U}_a$. So there must be some $\mathcal{B}_p \in \mathfrak{K}$ with $\mathcal{B}_p \not\models p$. This means that every $p \in \mathcal{U}_a$ is in some homomorphic image $\varphi_p[\mathcal{B}_p]$ of some $\mathcal{B}_p$ from $\mathfrak{K}$. Consequently, $\mathcal{U}_a$ is a subcoalgebra of the Union $\bigcup_{p\in\mathcal{U}} \varphi_p[\mathcal{B}_p] \leq \mathcal{T}_\kappa$. This shows that $\mathcal{A} \in \mathcal{SH}\Sigma(\mathfrak{K})$. □

So, coequations as sets of ***forbidden behavior*** patterns can be used to provide a logical language for coalgebras. Indeed, it is not hard to find a coequational calculus which is dual to Birkhoff's equational calculus. Given a set $P \subseteq \mathcal{T}_X$ of coequations, that is a set of forbidden patterns, a coalgebra $\mathcal{A}$ satisfies P if for each coloring $g : A \to X$ each homomorphic image $\tilde{g}[\mathcal{A}]$ is a subcoalgebra of $\mathcal{T}_X$ completely contained in the complement of P. Let $\mathcal{C}_P = [\mathcal{T}_X - P]$ be the largest coalgebra contained in the complement of P, then the consequences of P are given by $Con(P) = \mathcal{T}_X - [\mathcal{T}_X - P]$, see [6]. A set of rules for generating $Con(P)$ from P is presented in [3].

In spite of its mathematically pleasing duality to Birkhoff's theorem, it turns out that there are only few "natural" examples of previously studied classes of coalgebras - be they automata or Kripke structures. The main obstacle seems to be that coequations describe coalgebras globally: All elements must avoid a certain pattern. This is distinctly different in modal logics, which describe coalgebras locally.

5.6. Modal Logics.

The first successful attempt at designing a modal logic for coalgebras is due to L. Moss [18]. The formulas of his logical language were given as a fixpoint $\mathcal{L}_F$ of the equation $\mathcal{L}_F \cong \mathbb{P}(\mathcal{L}_F) + F(\mathcal{L}_F)$, so they always formed a proper class. To show expressiveness, it was necessary to assume that the type functor preserves weak pullbacks.

A different approach was championed in a series of papers by Pattinson [19] and Schröder [22], using so called *predicate transformers.* Essentially, a predicate transformer for a functor F is a transformation describing how to transform a predicate $R \subseteq X$ on a set X into a predicate $\mu(R) \subseteq F(X)$, more precisely, a natural transformation between the contra-variant functors 2^X and $2^{F(X)}$. Every such predicate transform gives rise to a modality, leading to a modal logic, which under mild conditions on the functor is both admissible with homomorphisms and expressive.

We present here a simplification of the Pattinson-Schröder approach, which, we believe, gives a rather concrete and intuitive interpretation to modalities.

5.7. Modalities as 0-1-pattern.

First recall that modal formulas in Kripke structures describe local properties as seen from a particular point s:

$s \models \Box\phi$	iff all immediate successors of s satisfy ϕ
$s \models \Diamond\phi$	iff some immediate successors of s satisfies ϕ.

Considering the same from a different angle, let $\alpha(s)$ be the set of immediate successors of s. Let ϕ be a state formula and let $[\![\phi]\!] : A \to 2$ denote the characteristic function for the elements of A satisfying ϕ. Then $[\![\phi]\!][\alpha(s)]$ replaces each element of $\alpha(s)$ by its truth value, yielding a pattern in $\mathbb{P}(\{0,1\})$. Note that

$s \models \Box\phi$ iff $[\![\phi]\!][\alpha(s)] = \{1\}$
$s \models \Diamond\phi$ iff $[\![\phi]\!][\alpha(s)] = \{1\}$ or $[\![\phi]\!][\alpha(s)] = \{0,1\}$

In other words, modalities are 0-1-patterns that are allowed for the validity of a formula in the successor set.

5.8. **Syntax and semantics.** We shall define modalities as 0-1-patterns and their semantics as validity patterns for the successor $\alpha(s)$ of a state s:

Definition 18. [Syntax] Given a $\mathcal{S}et$-functor F, define the formulas ϕ of modal logic by the syntax

$$\begin{array}{lll} \phi \;::= & [p]\phi & \text{for each } p \in F(2) \\ \quad\;\; \mid & \phi_1 \wedge \phi_2 & \mid \neg\phi \mid \mathit{true} \end{array}$$

The semantics of the boolean operators true, $\wedge$, and $\neg$ is the obvious one. Before spelling out the semantics of the modalities, we can already highlight the semantic role of the boolean operators. Here we use the symbol $\approx$ to denote logical equivalence:

Lemma 19. *For each map $f : A/_{\approx} \to 2$ and for each finite subset $U \subseteq A$ there exists a formula ϕ_f such that $[\![\phi]\!]_f \circ \iota_U = f \circ \pi_{\approx} \circ \iota_U$.*

$$\begin{tikzcd} U \arrow[r, hook, "\iota_U"'] & A \arrow[r, two heads, "\pi_{\approx}"'] \arrow[rr, dashed, bend left, "{[\![\phi]\!]}"] & A/_{\approx} \arrow[r, "f"'] & 2 \end{tikzcd}$$

Proof. For $a, b \in A$ with $a \not\approx b$ there exist a formula $\phi_{a,b}$ with $a \models \phi_{a,b}$ and $b \not\models \phi_{a,b}$. Then $\phi_a := \bigwedge\{\phi_{a,b} \mid b \in U,\, a \not\approx b\}$ defines a (up to logical equivalence) relative to U. Now the claim is true for $\phi_f := \bigvee\{\phi_a \mid a \in U,\, f(\pi_{\approx}(a)) = 1\}$. □

Definition 20. [Semantics] Given an F-coalgebra $\mathcal{A} = (A, \alpha_{\mathcal{A}})$ and an element $a \in A$, we define $a \models [p]\phi :\iff (F\,[\![\phi]\!])(\alpha_{\mathcal{A}}(a)) = p$.

This formulates precisely our intuition: Take the successor $\alpha_{\mathcal{A}}(a)$ of a and replace every occurrence of an element $x \in A$ by its logical value $[\![\phi]\!](x)$. We obtain a 0-1-pattern $p = F\,[\![\phi]\!](\alpha_{\mathcal{A}}(a)) \in F(2)$. Now $a \models [p]\phi$ means that "p is the validity pattern under ϕ of the successor of a".

$$\begin{tikzcd} {} \arrow[r, "a"] & A \arrow[r, "{[\![\phi]\!]}"] \arrow[d, "\alpha_{\mathcal{A}}"'] & 2 & \\ & F(A) \arrow[r, "{F\,[\![\phi]\!]}"] & F(2) & {} \arrow[l, "p"'] \end{tikzcd}$$

Observe, that the logic is such that we cannot have both $s \models [p]\phi$ and $s \models [q]\phi$ at the same time, unless $p = q$.

We discuss two examples:

- ▷ Assume that the functor F associates with any set X the set of all binary trees whose leaves are labeled with elements of X. A map $f : X \to Y$ is taken to a substitution map $Ff : F(X) \to F(Y)$ which in every tree of $F(X)$ replaces each leaf label x by the new label $f(x)$. Assume that in the F-coalgebra $\mathcal{A} = (A, \alpha)$ an element $a \in A$ is mapped by α to a tree $\alpha(a) = (a_1, (a_2, a_3))$,

$$\begin{tikzcd}
 & & \circ \arrow[dl, dash] \arrow[dr, dash] & & \\
a \arrow[r, mapsto, "\alpha_{\mathcal{A}}"] & a_1 & & \circ \arrow[dl, dash] \arrow[dr, dash] & \\
 & & a_2 & & a_3
\end{tikzcd}$$

  and that $a_1 \models \phi$, $a_3 \models \phi$, but $a_2 \not\models \phi$. Then $a \models [(1, (0, 1))]\phi$. So, $p = (1, (0, 1)) \in F(2)$ is the validity pattern of ϕ in the successor of a.
- ▷ Consider the finite distribution functor $\mathbb{D}$ which associates to each set X the set of all discrete probability distributions, i.e. all $\sigma \in [0, 1]^X$ with $\sigma(x) = 0$ for almost all $x \in X$ and $\Sigma_{x \in X}\sigma(x) = 1$. A map $f : X \to Y$ is taken to $\mathbb{D}f : \mathbb{D}(X) \to \mathbb{D}(Y)$ where $(\mathbb{D}f)(\sigma)(y) = \Sigma_{f(x)=y}\sigma(x)$. A 0-1-pattern $\sigma \in \mathbb{D}(2)$ for this functor is described uniquely by a number $p \in [0, 1]$, where $\sigma(1) = p$ and $\sigma(0) = 1 - p$.
  A $\mathbb{D}$-coalgebra $\mathcal{A} = (A, \alpha)$ can be interpreted as a probabilistic transition system, where for each $s \in A$ the distribution $\sigma = \alpha(s) \in \mathbb{D}(A)$ specifies for each other state s' the probability $\sigma(s')$ for it to be selected next.
  Given a state formula ϕ and $p \in [0, 1]$, the modality $[p]\phi$ describes the probability that the successor of s chosen next will satisfy ϕ. It can be computed as $\Sigma_{a \models \phi}\sigma(a)$ where $\sigma = \alpha(s)$. Therefore, $s \models [p]\phi \iff \Sigma_{a \models \phi}\alpha(s)(a) = p \iff \mathbb{D}\, [\![\phi]\!](\alpha(s)) = p$.

5.9. **Admissibility and expressiveness.** We intend to show that logical equivalence is the same as observational equivalence, in short: $\approx = \nabla$. The following lemma ensures the logic to be compatible with homomorphisms, we say that the logic is admissible:

Lemma 21. *If $\varphi : \mathcal{A} \to \mathcal{B}$ is a homomorphism and $a \in A$, then for any modal formula ϕ we have $a \models \phi \iff \varphi(a) \models \phi$.*

Proof. By induction on the construction of modal formulas. Passing from a formula ϕ to the modality $[p]\phi$ is the only interesting case.

By induction hypothesis, $a \models \phi \iff \varphi(a) \models \phi$, which means that the upper triangle in the figure commutes. Applying F yields the lower triangle, so this commutes, too. Commutativity of the rectangle is equivalent to φ being a homomorphism. Now we read off the following figure that the successor of a has the same ϕ-validity pattern as the successor of $\varphi(a)$, meaning that for each p we have

$a \models [p]\phi \iff \varphi(a) \models [p]\phi.$

$$\begin{tikzcd}
 & & 2 & & \\
a \arrow[r, mapsto] & A \arrow[ur, "\llbracket\phi\rrbracket"] \arrow[rr, "\varphi"] \arrow[d, "\alpha"] & & B \arrow[ul, "\llbracket\phi\rrbracket"'] \arrow[d, "\beta"] & \\
 & FA \arrow[rr, "F\varphi"] \arrow[dr, "F\ \llbracket\phi\rrbracket"'] & & FB \arrow[dl, "F\ \llbracket\phi\rrbracket"] & \\
 & & F2 & & p \arrow[ll, mapsto]
\end{tikzcd}$$

□

As a consequence and with the help of Lemma 9, observational equivalence implies modal equivalence, i.e. $\nabla \subseteq \approx$. The converse inclusion amounts to expressivity of the logic. It is valid under mild conditions on the functor:

(1) F is finitary, and
(2) F is 2-separating.

Both requirements may be weakened. The first condition permits the logic to be finitary. If F is κ-ary, the logical language will need κ-ary conjunctions. The second condition permits to work with unary modalities in a 2-valued logic. If F was $\kappa-$separating for some $\kappa > 2$, then we could either pass to a logic with κ truth values, or we could use κ-ary modalities $[p](\phi_1, ..., \phi_k, ...)$. We leave these modifications to the reader.

Theorem 22. *If the functor F is finitary and 2-separating, then finitary coalgebraic modal logic is expressive.*

Proof. In order to prove $\approx \subseteq \nabla$, it suffices to turn $A/\approx$ into a coalgebra so that $\pi_\approx : A \to A/\approx$ is a homomorphism. This requires to complete the following diagram:

$$\begin{tikzcd}
A \arrow[r, two heads, "\pi_\approx"] \arrow[d, "\alpha"'] & A/\approx \arrow[d, dotted] \\
F(A) \arrow[r, two heads, "F\pi_\approx"] & F(A/\approx)
\end{tikzcd}$$

The completion exists, provided we can show for arbitrary $x, y \in A$ that assuming $\pi_\approx(x) = \pi_\approx(y)$, i.e. $x \approx y$, implies $u_x := (F\pi_\approx)(\alpha(x)) = (F\pi_\approx)(\alpha(y)) =: u_y$.

By way of contradiction assume that $u_x \neq u_y$, then by 2-separability of F, there exists a map $f : A/\approx \to 2$ with $p_x := (Ff)(u_x) \neq (Ff)(u_y) =: p_y$. If we can find a formula ϕ_f so that $f \circ \pi_\approx = \llbracket\phi_f\rrbracket$, then $x \models [p_x]\phi_f$, whereas $y \models [p_y]\phi_f$, contradicting $x \approx y$.

$$\begin{tikzcd}
 & \bullet \arrow[d, shift right, "x"'] \arrow[d, shift left, "y"] & & \\
U \arrow[r, hook, "\iota_U"] & A \arrow[r, two heads, "\pi_\approx"] \arrow[d, "\alpha"] \arrow[rr, dashed, bend left, "\llbracket\phi_f\rrbracket"] & A/\approx \arrow[r, "f"] \arrow[d, dotted] & 2 \\
F(U) \arrow[r, hook, "F\iota_U"] & F(A) \arrow[r, "F\pi_\approx"] \arrow[rr, dashed, bend right, "F\ \llbracket\phi_f\rrbracket"'] & F(A/\approx) \arrow[r, "Ff"] & F(2) \\
 & \bullet \arrow[u, shift left, "\alpha(x)"] \arrow[u, shift right, "\alpha(y)"'] & \bullet \arrow[u, shift left, "u_x"] \arrow[u, shift right, "u_y"'] & \bullet \arrow[u, shift left, "p_x"] \arrow[u, shift right, "p_y"']
\end{tikzcd}$$

Using the fact that F is finitary, so we can find some finite $U \subseteq A$ with $\alpha(x), \alpha(y) \in (F\iota_U)[F(U)]$. Lemma 19 now provides a formula ϕ_f with $[\![\phi]\!]_f \circ \iota_U = f \circ \pi_{\approx} \circ \iota_U$. Applying F yields commutativity of the bottom row: $F\ [\![\phi]\!]_f \circ F\iota_U = Ff \circ F\pi_{\approx} \circ F\iota_U$. In particular, $F\ [\![\phi]\!]_f(\alpha(x)) = Ff \circ F\pi_{\approx}(\alpha(x)) = p_x$ and $F\ [\![\phi]\!]_f(\alpha(y)) = Ff \circ F\pi_{\approx}(\alpha(y)) = p_y$, so $x \models [p_x]\phi_f$ whereas $y \models [p_y]\phi_f$.

We may enrich the logic by allowing as modalities sets of patterns $P \subseteq F(2)$ rather than elements $p \in F(2)$. Obviously, $s \models [P]\phi$ is to mean $s \models [p]\varphi$ for some $p \in P$. Clearly, the logic remains admissible and expressive, and it may be preferable for practical purposes. Whereas the classical box-modality of Kripke structures is expressible either way: $s \models \Box\phi \iff s \models [\{\{\}, \{1\}\}]\phi \iff (s \models [\{\}]\phi \vee s \models [\{1\}]\phi)$, the added expressiveness is useful for probabilistic systems: $s \models [[0,p]]\phi$ means that the probability of a successor of s satisfying ϕ is between 0 and p, i.e. at most p. $\Box$

6. Conclusion

We have presented universal coalgebras as models for state based systems, looked at their basic structure theory and at several logics. The first logic amounts to a coinduction principle for terminal coalgebras, in particular for terminal data types. The second logic is based on coequations as elements of cofree coalgebras. For this logic we presented a Birkoff-style theorem. Modal logic, in contrast serves to describe coalgebras from a local perspective. We exposed an elegant proof of a correctness and completeness theorem which encompasses the famous Hennessy-Milner theorem for image finite Kripke structures.

Even though most of the the material has appeared in the literature, this is not the case for the intuition that we make explicit both regarding set functors and modalities. In hindsight, and with this intuition in mind, one wonders why much of this material had not been developed some 10 years earlier.

References

[1] P. Aczel and N. Mendler. A final coalgebra theorem. In D.H. Pitt et al., editors, *Proceedings category theory and computer science*, Lecture Notes in Computer Science, pages 357–365. Springer, 1989.

[2] J. Adámek, H. Herrlich, and G.E. Strecker. *Abstract and Concrete Categories.* John Wiley & Sons, 1990.

[3] J. Adámek. A logic of coequations. In *Proceedings CSL*, volume 3634 of *Lecture Notes in Computer Science*, pages 70–86. Springer, 2005.

[4] G. Grätzer. *Universal Algebra.* Springer Verlag, 1979.

[5] H.P. Gumm. Elements of the general theory of coalgebras. LUATCS 99, Rand Afrikaans University, Johannesburg, South Africa, 1999.

[6] H.P. Gumm. Birkhoff's variety theorem for coalgebras. *Contributions to General Algebra*, 13:159–173, 2000.

[7] H.P. Gumm. Functors for coalgebras. *Algebra Universalis*, (45 (2-3)):135–147, 2001.

[8] H.P. Gumm. On minimal coalgebras. *Applied Categorical Structures*, (16):313–332, 2008.

[9] H.P. Gumm, J. Hughes, and T. Schröder. Distributivity of categories of coalgebras. Technical report, 2003.

[10] H.P. Gumm and T. Schröder. Coalgebraic structure from weak limit preserving functors. *Electronic Notes in Theoretical Computer Science*, 33:113–133, 2000.

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

[12] H.H. Hansen, C. Kupke, and E. Pacuit. Bisimulations for neighbourhood structures. In *Proceedings of 2nd Conference on Algebra and Coalgebra in Computer Science*, Lecture Notes in Computer Science, 2007.

[13] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32, 1985.

[14] A. Igarashi, B.C. Pierce, and Ph. Wadler. Featherweight java: a minimal core calculus for java and gj. *ACM Trans. Program. Lang. Syst.*, 23(3):396–450, 2001.

[15] Ihringer and Gumm. *Allgemeine Algebra. Mit einem Anhang über Universelle Coalgebra.* Heldermann Verlag, 2003.

[16] A. Kurz. *Logics for Coalgebras and Applications to Computer Science.* PhD thesis, LMU München, 2000.

[17] J. Lambek. Least fixpoints of endofunctors of cartesian closed categories. *Mathematical Structures in Computer Science*, 3:229–257, 1993.

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

[19] D. Pattinson. Expressive logics for coalgebras via terminal sequence induction. *Notre Dame J. Formal Logic*, 45:19–33, 2004.

[20] J.J.M.M. Rutten. Automata and coinduction (an exercise in coalgebra). In D. Sangiorigi and R. de Simone, editors, *Proceedings of CONCUR '98*, number 1466 in LNCS, pages 194–218, 1998.

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

[22] L. Schröder. Expressivity of coalgebraic modal logic: The limits and beyond. *Theoretical Computer Science*, 390:230–247, 2008.

[23] V. Trnková. Some properties of set functors. *Comm. Math. Univ. Carolinae*, 10(2):323–352, 1969.

Philipps-Universität Marburg
E-mail address: gumm@mathematik.uni-marburg.de
URL: http://www.mathematik.uni-marburg.de/~gumm