Coalgebraic trace semantics via forgetful logics

Bartek Klin[1,⋆] and Jurriaan Rot[2,⋆⋆]

[1] University of Warsaw
[2] Leiden University

Abstract. We use modal logic as a framework for coalgebraic trace semantics, and show the flexibility of the approach with concrete examples such as the language semantics of weighted, alternating and tree automata. We provide a sufficient condition under which a logical semantics coincides with the trace semantics obtained via a given determinization construction. Finally, we consider a condition that guarantees the existence of a canonical determinization procedure that is correct with respect to a given logical semantics. That procedure is closely related to Brzozowski's minimization algorithm.

1 Introduction

Coalgebraic methods [23, 12] have been rather successful in modeling branching time behaviour of various kinds of transition systems, with a general notion of bisimulation and final semantics as the main contributions. Coalgebraic modeling of linear time behaviour such as trace semantics of transition systems or language semantics of automata, has also attracted significant attention. However, the emerging picture is considerably more complex: a few approaches have been developed whose scopes and connections are not yet fully understood. Here, we exacerbate the situation by suggesting yet another approach.

To study trace semantics coalgebraically, one usually considers systems whose behaviour type is a composite functor of the form TB or BT, where T represents a branching aspect of behaviour that trace semantics is supposed to "resolve", and B represents the transition aspect that should be recorded in system traces. Typically it is assumed that T is a monad, and its multiplication structure is used to resolve branching. For example, in [22, 10], a distributive law of B over T is used to lift B to the Kleisli category of T, and trace semantics is obtained as final semantics for the lifted functor. Additional assumptions on T are needed for this, so this approach does not work for coalgebras such as weighted automata. On the other hand, in [13, 25] a distributive law of T over B is used to lift B to the Eilenberg-Moore category of T, with trace semantics again obtained as final semantics for the lifted functor. This can be seen as a coalgebraic generalization of the powerset determinization procedure for non-deterministic

⋆ Supported by the Polish National Science Centre (NCN) grant 2012/07/E/ST6/03026.
⋆⋆ Supported by NWO project 612.063.920. This research was carried out during the second author's stay at University of Warsaw, partially supported by WCMCS.

automata. While it applies to many examples, that approach does not work for systems that do not determinize, such as tree automata. A detailed comparison of these two approaches is in [13]. In the recent [18], the entire functor TB (or BT) is embedded in a single monad, which provides some more flexibility. In [9], it is embedded in a more complex functor with a so-called observer.

In this paper, we study trace semantics in terms of modal logic. The basic idea is very simple: we view traces as formulas in suitable modal logics, and trace semantics of a state arises from all formulas that hold for it. A coalgebraic approach to modal logic based on dual adjunctions is by now well developed [21, 16, 14, 17], and we apply it to speak of traces generally. Obviously not every logic counts as a trace logic: assuming a behaviour type of the form BT or TB, we construct logics from arbitrary (but usually expressive) logics for B and special logics for T whose purpose is to resolve branching. We call such logics *forgetful*.

Our approach differs from previous studies in a few ways:

- We do not assume that T is a monad, unless we want to relate our logical approach to ones that do, in particular to determinization constructions.
- Instead of using monad multiplication $\mu\colon TT \Rightarrow T$ to resolve branching, we use a natural transformation $\alpha\colon TG \Rightarrow G$, where G is a contravariant functor that provides the basic infrastructure of logics. In case of nondeterministic systems, T is the covariant powerset functor and G the contravariant powerset, so TT and TG act the same on objects, but they carry significantly different intuitions.
- Trace semantics is obtained not as final semantics of coalgebras, but by initial semantics of algebras. Fundamentally, we view trace semantics as an inductive concept and not a coinductive one akin to bisimulation, although in some well-behaved cases the inductive and coinductive views coincide.
- Thanks to the flexibility of modal logics, we are able to cover examples such as the language semantics of weighted tree automata, that does not quite fit into previously studied approaches, or alternating automata.

The idea of using modal logics for coalgebraic trace semantics is not new; it is visible already in [21]. In [10] it is related to behavioural equivalence, and applied to non-deterministic systems. A generalized notion of relation lifting is used in [5] to obtain infinite trace semantics, and applied in [6] to get canonical linear time logics. In [15], coalgebraic modal logic is combined with the idea of lifting behaviours to Eilenberg-Moore categories, with trace semantics in mind. In [13], a connection to modal logics is sketched from the perspective of coalgebraic determinization procedures. In a sense, this paper describes the same connection from the perspective of logic.

Our main new contribution is the notion of forgetful logic and its ramifications. The basic definitions are provided in Section 3 and some illustrative examples in Section 4. We introduce a systematic way of relating trace semantics to determinization, by giving sufficient conditions for a given determinization procedure, understood in a slightly more general way than in [13], to be correct with respect to a given forgetful logic (Section 6). For instance, this allows showing in a coalgebraic setting that the determinization of alternating automata into non-deterministic automata preserves language semantics.

A correct determinization procedure may not exist in general. In Section 7 we study a situation where a canonical correct determinization procedure exists. It turns out that even in the simple case of non-deterministic automata that procedure is not the classical powerset construction; instead, it relies on a double application of contravariant powerset construction. Interestingly, this is what also happens in Brzozowski's algorithm for automata minimization [4], so as a by-product, we get a new perspective on that algorithm which has recently attracted much attention in the coalgebraic community [1–3].

Acknowledgments. We are grateful to Marcello Bonsangue, Helle Hvid Hansen, Ichiro Hasuo and Jan Rutten for discussions and suggestions, and to anonymous referees for their insightful comments.

2 Preliminaries

We assume familiarity with basic notions of category theory (see, e.g., [20]). A coalgebra for a functor $B\colon \mathcal{C} \to \mathcal{C}$ consists of an object X and a map $f\colon X \to BX$. A homomorphism from $f\colon X \to BX$ to $g\colon Y \to BY$ is a map $h\colon X \to Y$ such that $g \circ h = Bh \circ f$. The category of B-coalgebras is denoted $\mathsf{Coalg}(B)$. Algebras for a functor L are defined dually; the category of L-algebras and homomorphisms is denoted $\mathsf{Alg}(L)$.

We list a few examples, where $\mathcal{C} = \mathsf{Set}$, the category of sets and functions. Consider the functor $\mathcal{P}_\omega(A \times -)$, where $\mathcal{P}_\omega$ is the finite powerset functor and A is a fixed set. A coalgebra $f\colon X \to \mathcal{P}_\omega(A \times X)$ is a finitely branching labelled transition system: it maps every state to a finite set of next states. Coalgebras for the functor $(\mathcal{P}_\omega -)^A$ are image-finite labelled transition systems, i.e., the set of next states for every label is finite. When A is finite the two notions coincide. A coalgebra $f\colon X \to \mathcal{P}_\omega(A \times X + 1)$, where $1 = \{*\}$ is a singleton, is a non-deterministic automaton; a state x is accepting whenever $* \in f(x)$.

Consider the functor $BX = 2 \times X^A$, where 2 is a two-element set of truth values. A coalgebra $\langle o, f\rangle\colon X \to BX$ is a deterministic automaton; a state x is accepting if $o(x) = \mathtt{tt}$, and $f(x)$ is the transition function. The composition $B\mathcal{P}_\omega$ yields non-deterministic automata, presented in a different way than above. We shall also consider $B\mathcal{P}_\omega\mathcal{P}_\omega$-coalgebras, which represent a general version of alternating automata.

Let $\mathbb{S}$ be a semiring. Define $\mathcal{M}X = \{\varphi \in \mathbb{S}^X \mid \mathrm{supp}(\varphi) \text{ is finite}\}$ where $\mathrm{supp}(\varphi) = \{x \mid \varphi(x) \neq 0\}$, and $\mathcal{M}(f\colon X \to Y)(\varphi)(y) = \sum_{x \in f^{-1}(y)} \varphi(x)$. A *weighted automaton* is a coalgebra for the functor $\mathcal{M}(A \times - + 1)$. Let Σ be a polynomial functor corresponding to an algebraic signature. A *top-down weighted tree automaton* is a coalgebra for the functor $\mathcal{M}\Sigma$. For $\mathbb{S}$ the Boolean semiring these are *non-deterministic tree automata*. Similar to non-deterministic automata above, one can present weighted automata as coalgebras for $\mathbb{S} \times (\mathcal{M}-)^A$.

We note that $\mathcal{P}_\omega$ is a monad, by taking $\eta_X(x) = \{x\}$ and μ to be union. More generally, the functor $\mathcal{M}$ extends to a monad, by taking $\mu_X(\varphi)(x) =$

$\sum_{\psi\in\mathbb{S}^X}\varphi(\psi)\cdot\psi(x)$. The case of $\mathcal{P}_\omega$ is obtained by taking the Boolean semiring. Notice that the finite support condition is required for μ to be well-defined.

2.1 Contravariant adjunctions

The basic framework of coalgebraic logic is formed of two categories $\mathcal{C}$, $\mathcal{D}$ connected by functors $F\colon \mathcal{C}^{op}\to\mathcal{D}$ and $G\colon\mathcal{D}^{op}\to\mathcal{C}$ that form an adjunction $F^{op}\dashv G$. For example, one may take $\mathcal{C}=\mathcal{D}=\mathsf{Set}$ and $F=G=2^{-}$, for 2 a two-element set of logical values. The intuition is that objects of $\mathcal{C}$ are collections of processes, or states, and objects of $\mathcal{D}$ are logical theories.

To avoid cluttering the presentation with too much of the $(-)^{op}$ notation, we opt to treat F and G as *contravariant functors*, i.e., ones that reverse the direction of all arrows (maps), between $\mathcal{C}$ and $\mathcal{D}$. The adjunction then becomes a contravariant adjunction "on the right", meaning that there is a natural bijection

$$\mathcal{C}(X,G\Phi)\cong\mathcal{D}(\Phi,FX)\qquad\text{for } X\in\mathcal{C},\Phi\in\mathcal{D}.$$

Slightly abusing the notation, we shall denote both sides of this bijection by $(-)^\flat$. Applying the bijection to a map is referred to as transposing the map.

In such an adjunction, GF is a monad on $\mathcal{C}$, whose unit we denote by $\iota\colon\mathsf{Id}\Rightarrow GF$, and FG is a monad on $\mathcal{D}$, with unit denoted by $\epsilon\colon\mathsf{Id}\Rightarrow FG$. Both F and G map colimits to limits, by standard preservation results for adjoint functors.

In what follows, the reader need only remember that F and G are contravariant, i.e., they reverse maps and natural transformations. All other functors, except a few that lift F and G to other categories, are standard covariant functors.

3 Forgetful logics

We begin by recalling an approach to coalgebraic modal logic based on contravariant adjunctions, see, e.g., [16, 14]. Consider categories $\mathcal{C}$, $\mathcal{D}$ and functors F, G as in Section 2.1. Given an endofunctor $B\colon\mathcal{C}\to\mathcal{C}$, a *coalgebraic logic* to be interpreted on B-coalgebras is built of *syntax*, i.e., an endofunctor $L\colon\mathcal{D}\to\mathcal{D}$, and *semantics*, a natural transformation $\rho\colon LF\Rightarrow FB$. We will usually refer to ρ simply as a logic. If an initial L-algebra $a\colon L\Phi\to\Phi$ exists then, for any B-coalgebra $h\colon X\to BX$, the *logical semantics* of ρ on h is a map $s^\flat\colon X\to G\Phi$ obtained by transposing the map defined by initiality of a as on the left:

$$\begin{array}{ccc} L\Phi & \xrightarrow{Ls} & LFX \\ & & \downarrow{\scriptstyle \rho_X} \\ \downarrow{\scriptstyle a} & & FBX \\ & & \downarrow{\scriptstyle Fh} \\ \Phi & \xrightarrow[s]{} & FX \end{array} \qquad\qquad \begin{array}{ccc} \mathsf{Coalg}(B) & \xrightarrow{\hat{F}} & \mathsf{Alg}(L) \\ \downarrow & & \downarrow \\ \mathcal{C} & \xrightarrow[F]{} & \mathcal{D} \end{array} \qquad (1)$$

The mapping of a B-coalgebra $h\colon X\to BX$ to an L-algebra $Fh\circ\rho_X\colon LFX\to FX$ determines a contravariant functor $\hat{F}$ that lifts F, i.e., acts as F on carriers, depicted on the right above. This functor has no (contravariant) adjoint in general; later in Section 7 we shall study well-behaved situations when it does. Notice that $\hat{F}$ maps coalgebra homomorphisms to algebra homomorphisms, and indeed the logical semantics factors through coalgebra homomorphisms, i.e., behavioural equivalence implies logical equivalence. The converse holds if ρ is *expressive*, meaning that the logical semantics decomposes as a coalgebra homomorphism followed by a mono.

Example 1. Let $\mathcal{C} = \mathcal{D} = \mathsf{Set}$, $F = G = 2^{-}$, $B = 2 \times -^{A}$ and $L = A \times - + 1$. The initial algebra of L is the set A^* of words over A. We define a logic $\rho\colon LF \Rightarrow FB$ as follows: $\rho_X(*)(o,t) = o$ and $\rho_X(a,\varphi)(o,t) = \varphi(t(a))$. For a coalgebra $\langle o, f\rangle\colon X \to 2 \times X^A$ the logical semantics is a map $s^\flat\colon X \to 2^{A^*}$, yielding the usual language semantics of the automaton: $s^\flat(x)(\varepsilon) = o(x)$ for the empty word ε, and $s^\flat(x)(aw) = s^\flat(f(x)(a))(w)$ for any $a \in A, w \in A^*$.

Note that logical equivalences, understood as kernel relations of logical semantics, are conceptually different from behavioural equivalences typically considered in coalgebra theory, in that they do not arise from finality of coalgebras, but rather from initiality of algebras (albeit in a different category). Fundamentally, logical semantics for coalgebras is defined by induction rather than coinduction. In some particularly well-behaved cases the inductive and coinductive views coincide; we shall study such situations in Section 7.

A logic $\rho\colon LF \Rightarrow FB$ gives rise to its *mate* $\rho^\flat\colon BG \Rightarrow GL$, defined by

$$BG \xRightarrow{\iota BG} GFBG \xRightarrow{G\rho G} GLFG \xRightarrow{GL\epsilon} GL, \tag{2}$$

where ι and ϵ are as in Section 2.1. A routine calculation shows that ρ in turn is the mate of $\rho^\flat$ (with the roles of F, G, ι and ϵ swapped), giving a bijective correspondence between logics and their mates. Some important properties of logics are conveniently stated in terms of their mates; e.g., under mild additional assumptions (see [16]), if the mate is pointwise monic then the logic is expressive.

There is a direct characterization of logical semantic maps in terms of mates, first formulated in [21]. Indeed, by transposing (1) it is easy to check that the logical semantics $s^\flat\colon X \to G\Phi$ on a coalgebra $h\colon X \to BX$ is a unique map that makes the "twisted coalgebra morphism" diagram in (3) commute.

$$\begin{array}{ccc} BX & \xrightarrow{Bs^\flat} & BG\Phi \\ \uparrow{\scriptstyle h} & & \downarrow{\scriptstyle \rho^\flat_\Phi} \\ & & GL\Phi \\ & & \uparrow{\scriptstyle Ga} \\ X & \xrightarrow[s^\flat]{} & G\Phi. \end{array} \tag{3}$$

Logics for composite functors can often be obtained from logics of their components. Consider functors $B, T\colon \mathcal{C} \to \mathcal{C}$ and logics for them $\rho\colon LF \Rightarrow FB$ and $\alpha\colon NF \Rightarrow FT$, for some functors $L, N\colon \mathcal{D} \to \mathcal{D}$. One can then define logics for the functors TB and BT:

$$\alpha \circledcirc \rho = \alpha B \circ N\rho\colon NLF \Rightarrow FTB, \qquad \rho \circledcirc \alpha = \rho T \circ L\alpha\colon LNF \Rightarrow FBT.$$

It is easy to see that taking the mate of a logic respects this composition operator, i.e., that $(\alpha \circledcirc \rho)^\flat = \alpha^\flat \circledcirc \rho^\flat$. Such compositions of logics appear in [12] and were studied in a slightly more concrete setting in [7, 24].

We shall be interested in the case where the logic for T has a trivial syntax; in other words, where $N = \mathsf{Id}$. Intuitively speaking, we require a logic for T that consists of a single unary operator, which could therefore be elided in a syntactic presentation of logical formulas. The semantics of such an operator is defined by a natural transformation $\alpha\colon F \Rightarrow FT$ or equivalently by its mate $\alpha^\flat\colon TG \Rightarrow G$. Intuitively, the composite logics $\alpha \circledcirc \rho$ and $\rho \circledcirc \alpha$, when interpreted on TB- and BT-coalgebras respectively disregard, or forget, the aspect of their behaviour related to the functor T, in a manner prescribed by α. We call logics obtained in this fashion *forgetful logics.*

4 Examples

We instantiate the setting of Section 3 and use forgetful logics to obtain trace semantics for several concrete types of coalgebras: non-deterministic automata, transition systems, alternating automata and weighted tree automata.

In the first few examples we let $\mathcal{C} = \mathcal{D} = \mathsf{Set}$ and $F = G = 2^-$, and consider TB or BT-coalgebras, where $T = \mathcal{P}_\omega$ is the finite powerset functor. Our examples involve the logic $\alpha\colon 2^- \Rightarrow 2^{\mathcal{P}_\omega}$ defined by:

$$\alpha_X(\varphi)(S) = \mathsf{tt} \text{ iff } \exists x \in S.\varphi(x) = \mathsf{tt}. \tag{4}$$

This choice of F and G has been studied thoroughly in the field of coalgebraic logic, and our α is an example of the standard notion of predicate lifting [12, 17] corresponding to the so-called diamond modality. Its mate $\alpha^\flat\colon \mathcal{P}_\omega 2^- \Rightarrow 2^-$ is as follows: $\alpha^\flat_\Phi(S)(w) = \mathsf{tt}$ iff $\exists \varphi \in S.S(w) = \mathsf{tt}$. In all examples below, $\mathcal{P}_\omega$ could be replaced by the full powerset $\mathcal{P}$ without any problems.

Example 2. We define a forgetful logic $\alpha \circledcirc \rho$ for $\mathcal{P}_\omega B$, where $BX = A \times X + 1$; α is as above and ρ is given below in terms of its mate $\rho^\flat\colon BG \Rightarrow GL$, in such a way that the logical semantics yields the usual language semantics. We let $L = B$, hence A^* carries the structure of an initial L-algebra. As a result, the logical semantics on an automaton will be a map from states to languages (elements of 2^{A^*}). Define $\rho^\flat\colon A \times 2^- + 1 \Rightarrow 2^{A\times - +1}$ by

$$\rho^\flat_\Phi(*)(t) = \mathsf{tt} \text{ iff } t = * \qquad \rho^\flat_\Phi(a, \varphi)(t) = \mathsf{tt} \text{ iff } t = (a, w) \text{ and } \varphi(w) = \mathsf{tt},$$

for any set Φ. The semantics of the logic $\alpha \circledcirc \rho$ on an automaton $f\colon X \to \mathcal{P}_\omega BX$ is the map $s^\flat$ from (3), and it is easy to calculate that for any $x \in X$:

$$s^\flat(x)(\varepsilon) = \mathsf{tt} \text{ iff } * \in f(x),$$
$$s^\flat(x)(aw) = \mathsf{tt} \text{ iff } \exists y \in X.(a, y) \in f(x) \text{ and } s^\flat(y)(w) = \mathsf{tt},$$

for ε the empty word, and for all $a \in A$ and $w \in A^*$.

Note that the logic ρ in the above example is expressive. One may expect that given a different expressive logic θ involving the same functors, the forgetful logics

$\alpha \circledcirc \rho$ and $\alpha \circledcirc \theta$ yield the same logical equivalences, but this is not the case. For instance, define $\theta^\flat \colon BG \Rightarrow GL$ as $\theta^\flat_\Phi(*)(t) = \mathtt{tt}$ for all t, and $\theta^\flat_\Phi(a,\varphi) = \rho^\flat_\Phi(a,\varphi)$. This logic is expressive as well (since $\theta^\flat$ is componentwise monic) but in the semantics of the forgetful logic $\alpha \circledcirc \theta$, information on final states is discarded.

Example 3 (Length of words). The initial algebra of $LX = X + 1$ is $\mathbb{N}$, the set of natural numbers. Define a logic for $BX = A \times X + 1$ by its mate $\rho^\flat \colon A \times 2^- + 1 \Rightarrow 2^{-+1}$ as follows: $\rho^\flat_\Phi(*)(t) = \mathtt{tt}$ iff $t = *$, and $\rho^\flat_\Phi(a,\varphi)(t) = \mathtt{tt}$ iff $t = w$ and $\varphi(w) = \mathtt{tt}$. Note that this logic is not expressive. With the above α, we have a logic $\alpha \circledcirc \rho$, and given any $f \colon X \to \mathcal{P}_\omega(A \times X + 1)$, this yields $s^\flat \colon X \to 2^\mathbb{N}$ so that $s^\flat(x)(0) = \mathtt{tt}$ iff $* \in f(x)$ and $s^\flat(x)(n+1) = \mathtt{tt}$ iff $\exists a \in A, y \in X$ s.t. $(a, y) \in f(x)$ and $s^\flat(y)(n) = \mathtt{tt}$. Thus, $s^\flat(x)$ is the binary sequence which is $\mathtt{tt}$ at position n iff the automaton f accepts a word of length n, starting in state x.

Example 4 (Non-deterministic automata as BT-coalgebras). Consider the functor $BX = 2 \times X^A$. Let $LX = A \times X + 1$, let $\rho^\flat \colon 2 \times (2^-)^A \Rightarrow 2^{A \times - + 1}$ be the mate of the logic ρ given in Example 1; explicitly, it is the obvious isomorphism given by manipulating exponents:

$$\rho^\flat_\Phi(o,\varphi)(*) = o \qquad \rho^\flat_\Phi(o,\varphi)(a,w) = \varphi(a)(w) \tag{5}$$

The logical semantics $s^\flat \colon X \to 2^{A^*}$ of $\rho \circledcirc \alpha$ on a coalgebra $\langle o, f\rangle \colon X \to 2 \times \mathcal{P}_\omega(X)^A$ is the usual language semantics: for any $x \in X$ we have $s^\flat(x)(\varepsilon) = o(x)$, and $s^\flat(x)(aw) = \mathtt{tt}$ iff $s^\flat(y)(w) = \mathtt{tt}$ for some $y \in f(x)(a)$.

A minor variation on the above, taking $BX = X^A$ and adapting $\rho^\flat$ appropriately so that $\rho^\flat(t)(*) = \mathtt{tt}$ for any t, yields finite traces of transition systems.

Non-determinism can be resolved differently: in contrast to (4), consider $\beta^\flat \colon \mathcal{P}_\omega 2^- \Rightarrow 2^-$ given by $\beta^\flat_\Phi(S)(x) = \mathtt{tt}$ iff $\forall \varphi \in S. S(x) = \mathtt{tt}$. Similarly to (4), β is a predicate lifting that corresponds to the so-called box modality. The semantics $s^\flat$ induced by the forgetful logic $\rho \circledcirc \beta$ accepts a word if *all* paths end in an accepting state: $s^\flat(x)(\varepsilon) = o(x)$, and $s^\flat(x)(aw) = \mathtt{tt}$ iff $s^\flat(y)(w) = \mathtt{tt}$ for all $y \in f(x)(a)$. We call this the conjunctive semantics. In automata-theoretic terms, this is the language semantics for ($B\mathcal{P}_\omega$-coalgebras understood as) co-nondeterministic automata, i.e., alternating automata with only universal states.

Some non-examples. It is not clear how to use forgetful logics to give a conjunctive semantics to coalgebras for $\mathcal{P}_\omega(A \times X + -)$; simply using β together with ρ from Example 2 does not yield the expected logical semantics. Also, transition systems as $\mathcal{P}_\omega(A \times -)$-coalgebras do not work well; with α as in (4) the logical semantics of a state with no successors is always empty, while it should contain the empty trace.

Example 5 (Alternating automata). Consider $B\mathcal{P}_\omega\mathcal{P}_\omega$-coalgebras with $B = 2 \times -^A$. We give a forgetful logic by combining ρ, α, and β from the previous example (more precisely, the logic is $(\rho \circledcirc \alpha) \circledcirc \beta$); recall that α and β resolve the non-determinism by disjunction and conjunction respectively. Spelling out the details for a coalgebra $\langle o, f\rangle \colon X \to 2 \times (\mathcal{P}_\omega\mathcal{P}_\omega X)^A$ yields, for any $x \in X$: $s^\flat(x)(\varepsilon) = o(x)$ and for any $a \in A$ and $w \in A^*$: $s^\flat(x)(aw) = \mathtt{tt}$ iff there is $S \in f(x)(a)$ such that $s^\flat(y)(w) = \mathtt{tt}$ for all $y \in S$.

Example 6 (Weighted tree automata). In this example we let $\mathcal{C} = \mathcal{D} = \mathsf{Set}$ and $F = G = \mathbb{S}^-$ for a semiring $\mathbb{S}$. We consider coalgebras for $\mathcal{M}\Sigma$ (Section 2), where Σ is a polynomial functor corresponding to a signature. The initial algebra of Σ is carried by the set of finite Σ-trees, denoted by $\Sigma^*\emptyset$. Define $\rho\colon \Sigma F \Rightarrow F\Sigma$ by cases on the operators σ in the signature:

$$\rho_X(\sigma(\varphi_1,\dots,\varphi_n))(\tau(x_1,\dots,x_m)) = \begin{cases} \prod_{i=1..n} \varphi_i(x_i) & \text{if } \sigma = \tau \\ 0 & \text{otherwise} \end{cases}$$

where n is the arity of σ. Define $\alpha\colon \mathbb{S}^- \Rightarrow \mathbb{S}^{\mathcal{M}}$ by its mate: $\alpha^\flat_\Phi(\varphi)(w) = \sum_{\psi \in \mathbb{S}^\Phi} \varphi(\psi) \cdot \psi(w)$. Notice that α and ρ generalize the logics of Example 2.

Let $s^\flat$ be the logical semantics of $\alpha \circledcirc \rho$ on a weighted tree automaton $f\colon X \to \mathcal{M}\Sigma X$. For any tree $\sigma(t_1,\dots t_n)$ and any $x \in X$ we have:

$$s^\flat(x)(\sigma(t_1,\dots,t_n)) = \sum_{x_1,\dots,x_n \in X} f(x)(\sigma(x_1,\dots,x_n)) \cdot \prod_{i=1..n} s^\flat(x_i)(t_i)$$

As a special case, we obtain for any *weighted automaton* $f\colon X \to \mathcal{M}(A \times X + 1)$ a unique map $s^\flat\colon X \to \mathbb{S}^{A^*}$ so that for any $x \in X$, $a \in A$ and $w \in A^*$: $s^\flat(x)(\varepsilon) = f(x)(*)$ and $s^\flat(x)(aw) = \sum_{y \in X} f(x)(a,y) \cdot s^\flat(y)(w)$. For $\mathbb{S}$ the Boolean semiring we get the usual semantics of tree automata: $s^\flat(x)(\sigma(t_1,\dots,t_n)) = \mathsf{tt}$ iff there are $x_1,\dots,x_n$ such that $\sigma(x_1,\dots,x_n) \in f(x)$ and for all $i \le n$: $s^\flat(x_i)(t_i) = \mathsf{tt}$.

Notice that the Σ-algebra $\hat{F}(X,f)$ (see (1)) is a *deterministic bottom-up tree automaton*. It corresponds to the top-down automaton f, in the sense that the semantics $s^\flat$ of f is the transpose of the unique homomorphism $s\colon \Sigma^*\emptyset \to \mathbb{S}^X$ arising by initiality; the latter is the usual semantics of bottom-up tree automata.

5 Forgetful logics for monads

In most coalgebraic attempts to trace semantics [5, 9, 13, 15, 18, 22], the functor T, which models the branching aspect of system behaviour, is assumed to be a monad. The basic definition of a forgetful logic is more relaxed in that it allows an arbitrary functor T but one may notice that in all examples in Section 4, T is a monad.

In coalgebraic approaches cited above, the structure of T is resolved using monad multiplication $\mu\colon TT \Rightarrow T$. Forgetful logics use transformations $\alpha\colon F \Rightarrow FT$ with their mates $\alpha^\flat\colon TG \Rightarrow T$ for the same purpose. If T is a monad, it will be useful to assume a few basic axioms analogous to those of monad multiplication:

Definition 1. *Let (T,η,μ) be a monad. A natural transformation $\alpha^\flat\colon TG \Rightarrow G$ is a (T)-*action *(on G) if $\alpha^\flat \circ \eta G = \mathsf{id}$ and $\alpha^\flat \circ T\alpha^\flat = \alpha^\flat \circ \mu G$, i.e., if each component of $\alpha^\flat$ is an Eilenberg-Moore algebra for T.*

Just as monads generalize monoids, monad actions on functors generalize monoid actions on sets. We shall use properties of monad actions to relate forgetful logics to the determinization constructions of [13] in Section 6. It is easy to check by hand that in all examples in Section 4, $\alpha^\flat$ is an action, but it also follows from the following considerations.

In some well-structured cases, one can search for a suitable α by looking at T-algebras in $\mathcal{C}$. We mention it only briefly and not explain the details, as it will not be directly used in the following.

If $\mathcal{C}$ has products, then for any object $V \in \mathcal{C}$ there is a contravariant adjunction as in Section 2.1, where: $\mathcal{D} = \mathsf{Set}$, $F = \mathcal{C}(-, V)$ and $G = V^-$, where V^X denotes the X-fold product of V in $\mathcal{C}$. (This adjunction was studied in [19] for the purpose of combining distributive laws.) By the Yoneda Lemma, natural transformations $\alpha \colon F \Rightarrow FT$ are in bijective correspondence with algebras $g \colon TV \to V$. Routine calculation shows that the mate $\alpha^\flat$ is a T-action if and only if the corresponding g is an Eilenberg-Moore algebra for T.

Alternatively, one may assume that $\mathcal{C} = \mathcal{D}$ is a symmetric monoidal closed category and $F = G = V^-$ is the internal hom-functor based on an object $V \in \mathcal{C}$. (This adjunction was studied in [16] in the context of coalgebraic modal logic.) If, additionally, the functor T is strong, then every algebra $g \colon TV \to V$ gives rise to $\alpha \colon F \Rightarrow FT$, whose components $\alpha_X \colon V^X \to V^{TX}$ are given by transposing:

$$TX \otimes V^X \xrightarrow{\text{strength}} T(X \otimes V^X) \xrightarrow{T(\text{application})} TV \xrightarrow{g} V$$

If T is a strong monad and g is an E-M algebra for T then $\alpha^\flat$ is a T-action.

If $\mathcal{C} = \mathcal{D} = \mathsf{Set}$ then both these constructions apply (and coincide). All examples in Section 4 fit in this special case. In this situation more can be said [13, 12]: the resulting contravariant adjunction can be factored through the category of Eilenberg-Moore algebras for T.

6 Determinization

The classical powerset construction turns a non-deterministic automaton into a deterministic one, with states of the former interpreted as singleton states in the latter. More generally, a determinization procedure of coalgebras involves a change of state space. We define it as follows:

Definition 2. *For a functor T, a (T)-*determinization procedure *of H-coalgebras consists of a natural transformation $\eta \colon \mathsf{Id} \Rightarrow T$, a functor K and a lifting of T:*

$$\begin{array}{ccc} \mathsf{Coalg}(H) & \xrightarrow{\bar{T}} & \mathsf{Coalg}(K) \\ \downarrow & & \downarrow \\ \mathcal{C} & \xrightarrow{T} & \mathcal{C} \end{array}$$

We will mostly focus on cases where $H = TB$ or $H = BT$, but in Section 7 we will consider situations where T is not directly related to H.

The classical powerset construction is *correct*, in the sense that the language semantics of a state x in a non-deterministic automaton coincides with the final semantics (the accepted language) of the singleton of x in the determinized automaton. At the coalgebraic level, we capture trace semantics by a forgetful logic. Then, a determinization procedure is correct if logical equivalence on the original system coincides with behavioural equivalence on the determinized system along η:

Definition 3. *A determinization procedure $(\bar{T}, \eta)$ of H-coalgebras is* correct *wrt. a logic for H if for any H-coalgebra (X, f) with logical semantics $s^\flat$:*

1. *$s^\flat$ factors through $h \circ \eta_X$, for any K-coalgebra homomorphism h from $\bar{T}(X, f)$.*
2. *there exists a K-coalgebra homomorphism h from $\bar{T}(X, f)$ and a mono m so that $s^\flat = m \circ h \circ \eta_X$.*

The first condition states that behavioural equivalence on the determinized system implies logical equivalence on the original system; the second condition states the converse.

In [13] a more specific kind of determinization was studied, arising from a natural transformation $\kappa \colon TB \Rightarrow KT$ and a monad (T, η, μ). A determinization procedure T^κ for TB-coalgebras maps any $f \colon X \to TBX$ to

$$T^\kappa(X, f) = (TX \xrightarrow{Tf} TTBX \xrightarrow{\mu_{BX}} TBX \xrightarrow{\kappa_X} KTX) \tag{6}$$

It is easy to see that this construction respects homomorphisms, so that this indeed yields a lifting. For examples see, e.g., [13] and the end of this section.

The same type of natural transformation can be used to determinize BT-coalgebras, by mapping any $f \colon X \to BTX$ to

$$T_\kappa(X, f) = (TX \xrightarrow{Tf} TBTX \xrightarrow{\kappa_{TX}} KTTX \xrightarrow{K\mu_X} KTX) \tag{7}$$

This is considered in [25, 13] for the case where $B = K$ and κ is a distributive law of monad over functor. Again, this conforms to Definition 2.

The following gives a sufficient condition for the logical semantics on TB or BT-coalgebras to coincide with a logical semantics on determinized K-coalgebras.

Theorem 1. *Suppose (T, η, μ) is a monad and there are α, ρ, κ as above and $\theta \colon LF \Rightarrow FK$ so that $\alpha^\flat$ is an action and the following diagram commutes:*

$$\begin{array}{ccccc} TBG & \xrightarrow{T\rho^\flat} & TGL & \xrightarrow{\alpha^\flat L} & GL \\ \Downarrow{\scriptstyle \kappa G} & & & & \| \\ KTG & \xrightarrow{K\alpha^\flat} & KG & \xrightarrow{\theta^\flat} & GL. \end{array}$$

Let $s^\flat$ be the semantics of $\alpha \circledcirc \rho$ on some coalgebra $f \colon X \to TBX$, and let $s^\flat_\theta$ be the semantics of θ on $T^\kappa(X, f)$ (see (6)). Then $s^\flat = s^\flat_\theta \circ \eta_X$.

The same holds for the determinization procedure T_κ (see (7)) for BT-coalgebras and the logic $\rho \circledcirc \alpha$.

This can be connected to behavioural equivalence if θ is expressive:

Corollary 1. *Let (T, η, μ), α, ρ, θ and κ be as in Theorem 1, and suppose that θ is an expressive logic. Then the determinization procedure T^κ of TB-coalgebras* (6) *is correct with respect to $\alpha \circledcirc \rho$, and the determinization procedure T_κ of BT-coalgebras* (7) *is correct with respect to $\rho \circledcirc \alpha$.*

To illustrate all this, we show that the determinization of weighted automata as given in [13] is correct with respect to weighted language equivalence. (There is no such result for tree automata, as they do not determinize [8].)

Example 7. Fix a semiring $\mathbb{S}$, let $B = A \times - + 1$ and $K = \mathbb{S} \times -^A$. Consider $\kappa\colon \mathcal{M}B \Rightarrow K\mathcal{M}$ defined as follows [13]: $\kappa_X(\varphi) = (\varphi(*), \lambda a.\lambda x.\varphi(a, x))$. This induces a determinization procedure $\mathcal{M}^\kappa$ as in (6), for weighted automata. Let $\alpha \circledcirc \rho$ be the forgetful logic for weighted automata introduced in Example 6, and recall that the logical semantics on a weighted automaton is the usual notion of acceptance of weighted languages. We use Corollary 1 to prove that the determinization procedure $\mathcal{M}^\kappa$ is correct with respect to $\alpha \circledcirc \rho$. To this end, consider the logic $\theta^\flat\colon \mathbb{S} \times (\mathbb{S}^-)^A \Rightarrow \mathbb{S}^{A\times - + 1}$ given by the isomorphism, similar to the logic in Example 4. Since $\theta^\flat$ is componentwise injective, θ is expressive. Moreover, $\alpha^\flat$ is an action (see Section 5). The only remaining condition is commutativity of the diagram in Theorem 1, which is a straightforward calculation. This proves correctness of the determinization $\mathcal{M}^\kappa$ with respect to the semantics of $\alpha \circledcirc \rho$.

Example 8. In [25] it is shown how to determinize non-deterministic automata of the form $B\mathcal{P}_\omega$, where $BX = 2 \times X^A$, based on $\kappa = \langle \kappa^o, \kappa^t \rangle\colon \mathcal{P}_\omega(2 \times -^A) \Rightarrow 2 \times (\mathcal{P}_\omega -)^A$ (note that $B = K$ in this example) where $\kappa^o_X(S) = \mathtt{tt}$ iff $\exists t.(\mathtt{tt}, t) \in S$, and $\kappa^t_X(a) = \{x \mid x \in t(a) \text{ for some } (o, t) \in S\}$. In Example 4 we have seen an expressive logic ρ and an α so that the logical semantics of $\rho \circledcirc \alpha$ yields the usual language semantics. It is now straightforward to check that the determinization κ together with the logics ρ, α above satisfies the condition of Theorem 1, where $\theta = \rho$. By Corollary 1 this shows the expected result that determinization of non-deterministic automata is correct with respect to language semantics.

Moreover, recall that the logic $\rho \circledcirc \beta$, where β is as defined in Example 4, yields a conjunctive semantics. Take the natural transformation $\tau = \langle \tau^o, \tau^t \rangle$ of the same type as κ, where $\tau^o(S) = \mathtt{tt}$ iff $o = \mathtt{tt}$ for every $(o, t) \in S$, and $\tau^t = \kappa^t$. Using Corollary 1 we can verify that this determinization procedure is correct.

One can also get the finite trace semantics of transition systems (Example 4) by turning them into non-deterministic automata (then, B and K are different).

Example 9. Alternating automata (Example 5) can be determinized into non-deterministic automata; we show that this determinization preserves language semantics, using Theorem 1. Notice that this does not involve final semantics.

Let ρ, α, β and τ be as in Example 8, and let $\chi\colon \mathcal{P}_\omega\mathcal{P}_\omega \Rightarrow \mathcal{P}_\omega\mathcal{P}_\omega$ be as follows: $\chi_X(S) = \{\overrightarrow{g}(S) \mid g\colon S \to X \text{ s.t. } g(U) \in U \text{ for each } U \in S\}$, that is, given a family of sets S, it returns all possible sets obtained by choosing one element from each set in S. Now the composition $B\chi \circ \tau\mathcal{P}_\omega : \mathcal{P}_\omega B \mathcal{P}_\omega \Rightarrow B\mathcal{P}_\omega\mathcal{P}_\omega$ yields a determinization procedure, turning an alternating automaton into a non-deterministic one over sets of states (to be interpreted as conjunctions). We instantiate Theorem 1 by $T = \mathcal{P}_\omega$, the functor B from the theorem is $BT = 2 \times T^A$, the logics ρ and θ are instantiated respectively to ρ and $\rho \circledcirc \alpha$ from above. Then commutativity of the diagram in Theorem 1 boils down to the similar diagram for τ given in Example 8, and that χ distributes conjunction over disjunction. Finally, $\beta^\flat$ is an action of the powerset monad (Section 5). By Theorem 1 we obtain that for any alternating automaton: $s^\flat = s^\flat_{\rho\circledcirc\alpha} \circ \eta_X$ where X is the set of states, $s^\flat$ is the semantics and $s^\flat_{\rho\circledcirc\alpha}$ is the usual language semantics on the non-deterministic automaton obtained by determinization.

7 Logics whose mates are isomorphisms

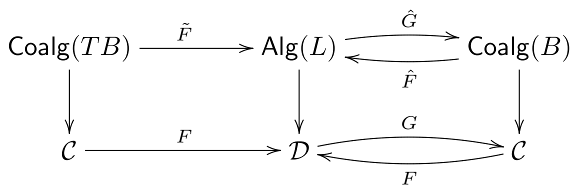
Corollary 1 provides a sufficient condition for a given determinization procedure to be correct with respect to a forgetful logic. However, in general there is no guarantee that a correct determinization procedure for a given logic exists. Indeed it would be quite surprising if it did: the language semantics of (weighted) tree automata (see Example 6) is an example of a forgetful logic, and such automata are well known not to determinize in a classical setting.

In this section we provide a sufficient condition for a correct determinization procedure to exist. Specifically, for an endofunctor B, we assume a logic ρ whose mate $\rho^\flat : BG \Rightarrow GL$ is a natural isomorphism. This condition holds, for instance, for ρ in Example 4 and for θ in Example 7. It has been studied before in the context of determinization constructions [13]. Its important consequence is that $s^\flat$ in (3) from Section 3 can be seen as a B-coalgebra morphism from (X, h) to $(G\Phi, (\rho^\flat_\Phi)^{-1} \circ Ga)$. Moreover, as shown in [13, Lemma 6] (see also [11]), the construction mapping any $g : LA \to A$ to $(\rho^\flat_A)^{-1} \circ Gg : GA \to BGA$ defines a functor $\hat{G} \colon \mathsf{Alg}(L) \to \mathsf{Coalg}(B)$, which is a contravariant adjoint to $\hat{F}$ (see (1) in Section 3). As a result, $\hat{G}$ maps initial objects to final ones, hence $(G\Phi, (\rho^\flat_\Phi)^{-1} \circ Ga)$ is a final B-coalgebra, therefore $s^\flat$ is a final coalgebra morphism from (X, h).

In the remainder of this section, due to space limitations we only deal with TB-coalgebras. However, a completely analogous development can be made for BT-coalgebras with little effort.

7.1 Canonical determinization

The setting of a forgetful logic $\alpha \circledcirc \rho$ where the mate of ρ is a natural isomorphism gives rise to the following diagram:



The functor $\tilde{F}$ arises from the logic $\alpha \circledcirc \rho$, the functor $\hat{F}$ arises from ρ and its contravariant adjoint $\hat{G}$ from the fact that $\rho^\flat$ is iso. Note that we make no assumptions on α; in particular, $\alpha^\flat$ need not be an action.

The composition $\hat{G}\tilde{F}$ is a determinization procedure, turning a coalgebra $f\colon X \to TBX$ into a B-coalgebra with carrier GFX. Explicitly, $\hat{G}\tilde{F}(X, f)$ is

$$GFX \xrightarrow{GFf} GFTBX \xrightarrow{G\alpha_{BX}} GFBX \xrightarrow{G\rho_X} GLFX \xrightarrow{(\rho^\flat)^{-1}_{FX}} BGFX \tag{8}$$

This determinization procedure is correct with respect to $\alpha \circledcirc \rho$ in the following sense, much stronger then required by Definition 3:

Theorem 2. *For any TB-coalgebra (X, f), the logical semantics $s^\flat$ of $\alpha \circledcirc \rho$ on (X, f) coincides with the final semantics of the B-coalgebra $\hat{G}\tilde{F}(X, f)$ precomposed with $\iota\colon \mathsf{Id} \Rightarrow GF$.*

Strictly speaking, this is not an example of a determinization procedure as understood in [13]: the functor $\hat{G}\tilde{F}$ lifts GF rather than T, and the lifting does not arise from a distributive law κ as described in Section 6. However, it is *almost* an example: after an encoding of TB-coalgebras as GFB-coalgebras, it arises from a distributive law $\kappa\colon GFB \Rightarrow BGF$.

Indeed, define $\Gamma\colon \mathsf{Coalg}(TB) \to \mathsf{Coalg}(GFB)$ by:

$$\Gamma(X, f) = (X, \gamma_{BX} \circ f) \qquad \text{where} \qquad \gamma = \alpha^\flat F \circ T\iota : T \Rightarrow GF. \tag{9}$$

GFB-coalgebras have a forgetful logic $\bar{\alpha} \circledcirc \rho$, where

$$\bar{\alpha} = \epsilon F : F \Rightarrow FGF, \qquad \text{equivalently,} \qquad \bar{\alpha}^\flat = G\epsilon : GFG \Rightarrow G.$$

(Note that $\bar{\alpha}^\flat$ is *always* a GF-action on G.) It is not difficult to calculate that for any TB-coalgebra (X, f), the logical semantics of $\bar{\alpha} \circledcirc \rho$ on $\Gamma(X, f)$ coincides with the logical semantics of $\alpha \circledcirc \rho$ on (X, f). Thus, encoding TB-coalgebras as GFB-coalgebras does not change their logical semantics.

Thanks to the mate $\rho^\flat\colon BG \Rightarrow GL$ being an isomorphism, the monad GF has a distributive law over B, denoted $\kappa\colon GFB \Rightarrow BGF$ and defined by:

$$GFB \xRightarrow{G\rho} GLF \xRightarrow{(\rho^\flat)^{-1}F} BGF \tag{10}$$

Using κ we can apply the determinization construction from [13] as described in Section 6, putting $K = B$. Straightforward diagram chasing using Corollary 1 shows that the determinization procedure $(GF)^\kappa$ defined as in (6) is correct with respect to $\bar{\alpha} \circledcirc \rho$. Altogether, a two-step determinization procedure arises:

$$\begin{array}{ccccc}
\mathsf{Coalg}(TB) & \xrightarrow{\Gamma} & \mathsf{Coalg}(GFB) & \xrightarrow{(GF)^\kappa} & \mathsf{Coalg}(B) \\
\downarrow & & \downarrow & & \downarrow \\
\mathcal{C} & \xrightarrow{\qquad \mathsf{Id} \qquad} & \mathcal{C} & \xrightarrow{\qquad GF \qquad} & \mathcal{C}
\end{array}$$

and it is correct with respect to $\alpha \circledcirc \rho$. Correctness can also be proved without Corollary 1, since the procedure coincides with the construction from (8):

Theorem 3. $(GF)^\kappa \circ \Gamma = \hat{G} \circ \tilde{F}$.

7.2 A connection to Brzozowski's algorithm

Call a B-coalgebra *observable* if the morphism into a final coalgebra (assuming it exists) is mono [3]. The above canonical determinization procedure can be adapted to construct, for any TB-coalgebra, an observable B-coalgebra whose final semantics coincides with the logical semantics on the original one.

Indeed, suppose $\mathsf{Alg}(L)$ has an (epi,mono)-factorization system. Given a coalgebra $f\colon X \to TBX$, the algebra homomorphism $s\colon (\varPhi, a) \to \tilde{F}(X,f)$ then decomposes as $s = m \circ e$, where m and e are mono and epi respectively; call the L-algebra in the middle (R,r). Recall that Gs is a coalgebra homomorphism into the final coalgebra. In the present situation it decomposes as follows:

$$\overbrace{\hat{G}\tilde{F}(X,f) \xrightarrow[Gm]{} \hat{G}(R,r) \xrightarrow[Ge]{} \hat{G}(\varPhi,a)}^{Gs}$$

and recall that $\hat{G}(\varPhi,a)$ is a final coalgebra. Because G is a right adjoint, it maps epis to monos, therefore Ge is mono and $\hat{G}(R,r)$ is observable. Moreover, thanks to Theorem 2 we have $s^\flat = Ge \circ Gm \circ \iota_X$, hence the final semantics Ge of $\hat{G}(R,r)$ coincides with the logical semantics on (X,f) along the mapping $Gm \circ \iota_X$.

Note that the construction of $\hat{G}(R,r)$ from (X,f) is not a determinization procedure itself according to Definition 2, as it does not lift any functor on $\mathcal{C}$.

The above refers to TB-coalgebras, but as everything else in this section, analogous reasoning works also for BT-coalgebras. For $T = \mathsf{Id}$ and $B = 2 \times -^A$, that (almost) corresponds to Brzozowski's algorithm for minimization of deterministic automata [4]. Applying $\tilde{F}$ to the given automaton corresponds to reversing transitions and turning final states into initial ones. Epi-mono factorization corresponds to taking the *reachable* part of this automaton. Then, applying $\hat{G}$ reverses transitions again, and turns initial states into final ones. Our abstract approach stops here; the original algorithm concludes by taking the reachable part again, which ensures minimality.

For a more detailed coalgebraic presentation of several concrete examples see [3]. Another approach, based on duality theory, is presented in [2]; this is related to the present development, but it uses dual equivalences rather than plain contravariant adjunctions. Another coalgebraic approach to minimization, based on factorization structures, is in [1]. A precise connection of these works to the present development is yet to be understood.

Notice that we only assume the mate of ρ to be iso; there are no requirements on α. The mate of ρ is iso for the logic from Example 4. Thus, we can instantiate α to obtain observable deterministic automata from non-deterministic automata or even alternating automata (by taking $T = \mathcal{P}_\omega\mathcal{P}_\omega$ and, for α, the composition of α and β from Example 5). The logic θ from Example 7 is covered as well, so one can treat Moore automata and weighted automata. However, the abstract construction of an observable automaton does not necessarily yield a concrete algorithm, as discussed for the case of weighted automata in [3].

References

1. J. Adámek, F. Bonchi, M. Hülsbusch, B. König, S. Milius, and A. Silva. A coalgebraic perspective on minimization and determinization. In *Procs. FOSSACS 2012*, volume 7213 of *LNCS*, pages 58–73, 2012.
2. N. Bezhanishvili, C. Kupke, and P. Panangaden. Minimization via duality. In *Procs. WoLLIC 2012*, volume 7456 of *LNCS*, pages 191–205, 2012.
3. F. Bonchi, M. M. Bonsangue, H. H. Hansen, P. Panangaden, J. J. M. M. Rutten, and A. Silva. Algebra-coalgebra duality in Brzozowski's minimization algorithm. *ACM Trans. Comput. Log.*, 15(1):3, 2014.
4. J. Brzozowski. Canonical regular expressions and minimal state graphs for definite events. *Mathematical Theory of Automata*, 12:529–561, 1962.
5. C. Cîrstea. From branching to linear time, coalgebraically. In *Procs. FICS 2013*, volume 126 of *EPTCS*, pages 11–27, 2013.
6. C. Cîrstea. A coalgebraic approach to linear-time logics. In *Procs. FOSSACS 2014*, volume 8412 of *LNCS*, pages 426–440, 2014.
7. C. Cîrstea and D. Pattinson. Modular construction of modal logics. In *Procs. CONCUR 2004*, volume 3170 of *LNCS*, pages 258–275, 2004.
8. H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: http://www.grappa.univ-lille3.fr/tata, 2007. release October, 12th 2007.
9. S. Goncharov. Trace semantics via generic observations. In *Procs. CALCO 2013*, volume 8089 of *LNCS*, pages 158–174, 2013.
10. I. Hasuo, B. Jacobs, and A. Sokolova. Generic trace semantics via coinduction. *Log. Meth. Comp. Sci.*, 3(4), 2007.
11. C. Hermida and B. Jacobs. Structural induction and coinduction in a fibrational setting. *Inf. and Comp.*, 145:107–152, 1997.
12. B. Jacobs. Introduction to coalgebra. Towards mathematics of states and observations, 2014. Draft.
13. B. Jacobs, A. Silva, and A. Sokolova. Trace semantics via determinization. *J. Comp. and Sys. Sci.*, 2014. To appear.
14. B. Jacobs and A. Sokolova. Exemplaric expressivity of modal logics. *J. Log. and Comput.*, 20(5):1041–1068, 2010.
15. C. Kissig and A. Kurz. Generic trace logics. *CoRR*, abs/1103.3239, 2011.
16. B. Klin. Coalgebraic modal logic beyond sets. *ENTCS*, 173:177–201, 2007.
17. C. Kupke and D. Pattinson. Coalgebraic semantics of modal logics: An overview. *Theor. Comput. Sci.*, 412(38):5070–5094, 2011.
18. A. Kurz, S. Milius, D. Pattinson, and L. Schröder. Simplified coalgebraic trace equivalence. *CoRR*, abs/1410.2463, 2014.
19. M. Lenisa, J. Power, and H. Watanabe. Category theory for operational semantics. *Theor. Comput. Sci.*, 327(1-2):135–154, 2004.
20. S. Mac Lane. *Categories for the working mathematician*, volume 5. Springer, 1998.
21. D. Pavlovic, M. Mislove, and J. Worrell. Testing semantics: Connecting processes and process logics. In *Procs. AMAST 2006*, volume 4019 of *LNCS*, pages 308–322, 2006.
22. J. Power and D. Turi. A coalgebraic foundation for linear time semantics. *ENTCS*, 29:259–274, 1999.
23. J. J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theor. Comput. Sci.*, 249(1):3–80, 2000.

24. L. Schröder and D. Pattinson. Modular algorithms for heterogeneous modal logics via multi-sorted coalgebra. *Math. Struct. in Comp. Sci.*, 21(2):235–266, 2011.
25. A. Silva, F. Bonchi, M. M. Bonsangue, and J. J. M. M. Rutten. Generalizing determinization from automata to coalgebras. *Log. Meth. Comp. Sci.*, 9(1), 2013.

A A useful lemma about mates

Note that standard counit-unit equations for adjunctions amount to:

$$\begin{array}{ccc} F & \xRightarrow{\epsilon F} & FGF \\ & \searrow^{=} & \Downarrow{\scriptstyle F\iota} \\ & & F \end{array} \qquad\qquad \begin{array}{ccc} G & \xRightarrow{\iota G} & GFG \\ & \searrow^{=} & \Downarrow{\scriptstyle G\epsilon} \\ & & G \end{array} \tag{11}$$

Two simple but useful diagrams show how logics relate to their mates along the basic adjunction:

Lemma 1. *For any logic $\rho\colon LF \Rightarrow FB$, the following diagrams commute:*

$$\begin{array}{ccc} B & \xRightarrow{\iota B} & GFB \\ {\scriptstyle B\iota}\Downarrow & & \Downarrow{\scriptstyle G\rho} \\ BGF & \xRightarrow[\rho^\flat F]{} & GLF \end{array} \qquad\qquad \begin{array}{ccc} L & \xRightarrow{\epsilon L} & FGL \\ {\scriptstyle L\epsilon}\Downarrow & & \Downarrow{\scriptstyle F\rho^\flat} \\ LFG & \xRightarrow[\rho G]{} & FBG. \end{array}$$

Proof. For the first diagram, chase:

$$\begin{array}{ccccccc} B & & & \xRightarrow{\iota B} & & & GFB \\ {\scriptstyle B\iota}\Downarrow & & & {\scriptstyle GFB\iota}\swarrow & & & \Downarrow{\scriptstyle G\rho} \\ BGF & \xRightarrow{\iota BGF} & GFBGF & \xRightarrow[G\rho GF]{} & GLFGF & \xLeftarrow{GLF\iota} & GLF \\ & & & & & {\scriptstyle GL\epsilon F}\searrow & \| \\ & & & \xRightarrow[\rho^\flat F]{} & & & GLF \end{array}$$

where everything commutes, clockwise starting from top-left: by naturality of ι, by naturality of ρ, by (11) above, and by definition of $\rho^\flat$.

The other diagram is similar. ◻

B Details of Section 4

In this section we expand on some of the examples considered in Section 4.

Example 2. The semantics of the logic $\alpha \circledcirc \rho$ on an automaton $f \colon X \to \mathcal{P}_\omega BX$ is the unique map $s^\flat$ making the following diagram commute (see (3)):

$$\begin{array}{ccccccc}
X & & & \xrightarrow{\quad s^\flat \quad} & & & 2^{A^*} \\
\downarrow{\scriptstyle f} & & & & & & \downarrow \\
\mathcal{P}_\omega(A \times X + 1) & \xrightarrow{\mathcal{P}_\omega B s^\flat} & \mathcal{P}_\omega(A \times 2^{A^*} + 1) & \xrightarrow{\mathcal{P}_\omega \rho^\flat_{A^*}} & \mathcal{P}_\omega(2^{A \times A^* + 1}) & \xrightarrow{\alpha^\flat_{LA^*}} & 2^{A \times A^* + 1}
\end{array}$$

and we spell out the details:

$$\begin{aligned}
s^\flat(x)(\varepsilon) = \mathsf{tt} \text{ iff } & (\alpha^\flat_{LA^*} \circ \mathcal{P}_\omega \rho^\flat_{A^*} \circ \mathcal{P}_\omega B s^\flat \circ f(x))(*) = \mathsf{tt} \\
\text{iff } & \exists \varphi \in (\mathcal{P}_\omega \rho^\flat_{A^*} \circ \mathcal{P}_\omega B s^\flat \circ f(x)).\varphi(*) = \mathsf{tt} \\
\text{iff } & \exists t \in f(x).(\rho^\flat_{A^*} \circ B s^\flat(t))(*) = \mathsf{tt} \\
\text{iff } & * \in f(x)
\end{aligned}$$

where ε is the empty word, and for all $a \in A$ and $w \in A^*$:

$$\begin{aligned}
s^\flat(x)(aw) = \mathsf{tt} \text{ iff } & \exists t \in f(x).(\rho^\flat_{A^*} \circ B s^\flat(t))(a, w) = \mathsf{tt} \\
\text{iff } & \exists t \in f(x).B s^\flat(t) = (a, \varphi) \text{ and } \varphi(w) = \mathsf{tt} \\
\text{iff } & \exists y \in X.(a, y) \in f(x) \text{ and } s^\flat(y)(w) = \mathsf{tt}
\end{aligned}$$

Example 5. The semantics of $\rho \circledcirc \alpha \circledcirc \beta$ on a coalgebra $\langle o, f \rangle \colon X \to B\mathcal{P}_\omega\mathcal{P}_\omega X$ is as follows (see (3)):

$$\begin{array}{ccccccccc}
X & & & & \xrightarrow{\quad s^\flat \quad} & & & & 2^{A^*} \\
\downarrow{\scriptstyle \langle o, f \rangle} & & & & & & & & \downarrow \\
B\mathcal{P}_\omega\mathcal{P}_\omega X & \xrightarrow{B\mathcal{P}_\omega\mathcal{P}_\omega s^\flat} & B\mathcal{P}_\omega\mathcal{P}_\omega 2^{A^*} & \xrightarrow{B\mathcal{P}_\omega \beta^\flat_{A^*}} & B\mathcal{P}_\omega 2^{A^*} & \xrightarrow{B\alpha^\flat_{A^*}} & B2^{A^*} & \xrightarrow{\rho^\flat_{A^*}} & 2^{LA^*}
\end{array} \tag{12}$$

Spelling this out yields $s^\flat(x)(\varepsilon) = o(x)$, and

$$\begin{aligned}
s^\flat(x)(aw) = \mathsf{tt} \text{ iff } & (\alpha^\flat_{A^*} \circ \mathcal{P}_\omega \beta^\flat_{A^*} \circ (\mathcal{P}_\omega\mathcal{P}_\omega s^\flat)(f(x)(a)))(w) = \mathsf{tt} \\
\text{iff } & \exists \varphi \in \mathcal{P}_\omega \beta^\flat_{A^*} \circ (\mathcal{P}_\omega\mathcal{P}_\omega s^\flat)(f(x)(a)) \text{ s.t. } \varphi(w) = \mathsf{tt} \\
\text{iff } & \exists U \in (\mathcal{P}_\omega\mathcal{P}_\omega s^\flat)(f(x)(a)) \text{ s.t. } \varphi(w) = \mathsf{tt} \text{ for all } \varphi \in U \\
\text{iff } & \exists S \in f(x)(a) \text{ s.t. } s^\flat(y)(w) = \mathsf{tt} \text{ for all } y \in f(x)(a)
\end{aligned}$$

Example 6. The logical semantics of this forgetful logic on a weighted tree automaton $f \colon X \to \mathcal{M}\Sigma X$ is the unique map $s^\flat \colon X \to \mathbb{S}^{\Sigma^*\emptyset}$ making the following diagram commute:

$$\begin{array}{ccccccc}
X & & & \xrightarrow{\quad s^\flat \quad} & & & \mathbb{S}^{\Sigma^*\emptyset} \\
\downarrow{\scriptstyle f} & & & & & & \downarrow \\
\mathcal{M}\Sigma X & \xrightarrow{\mathcal{M}\Sigma s^\flat} & \mathcal{M}\Sigma\mathbb{S}^{\Sigma^*\emptyset} & \xrightarrow{\mathcal{M}\rho^\flat_{\Sigma^*\emptyset}} & \mathcal{M}\mathbb{S}^{\Sigma\Sigma^*\emptyset} & \xrightarrow{\alpha^\flat_{\Sigma\Sigma^*\emptyset}} & \mathbb{S}^{\Sigma\Sigma^*\emptyset}
\end{array} \tag{13}$$

In order to understand this semantics, we first compute the composite logic $\alpha^\flat_\Sigma \circ \mathcal{M}\rho^\flat \colon \mathcal{M}\Sigma\mathbb{S}^- \Rightarrow S^\Sigma$:

$$\begin{aligned}(\alpha^\flat_{\Sigma\Phi} \circ \mathcal{M}\rho^\flat_\Phi(\varphi))(\sigma(w_1,\ldots,w_n)) &= \sum_{\psi\in\mathbb{S}^{\Sigma\Phi}} (\mathcal{M}\rho^\flat_\Phi(\varphi))(\psi)\cdot\psi(\sigma(w_1,\ldots,w_n)) \\ &= \sum_{\psi\in\mathbb{S}^{\Sigma\Phi}} \sum_{\gamma\in{\rho^\flat_\Phi}^{-1}(\psi)} \varphi(\gamma)\cdot\psi(\sigma(w_1,\ldots,w_n)) \\ &= \sum_{\varphi_1,\ldots,\varphi_n\in\mathbb{S}^\Phi} \varphi(\sigma(\varphi_1,\ldots,\varphi_n))\cdot\prod_{i=1..n}\varphi_i(w_i)\end{aligned}$$

The next step is to instantiate this to $\Sigma^*\emptyset$ and precompose with $\mathcal{M}\Sigma s^\flat$:

$$\begin{aligned}&(\alpha^\flat_{\Sigma\Sigma^*\emptyset} \circ \mathcal{M}\rho^\flat \circ \mathcal{M}\Sigma s^\flat(\psi))(\sigma(t_1,\ldots,t_n)) \\ &= \sum_{\varphi_1,\ldots,\varphi_n\in\mathbb{S}^{\Sigma^*\emptyset}} (\mathcal{M}\Sigma s^\flat(\psi))(\sigma(\varphi_1,\ldots,\varphi_n))\cdot\prod_{i=1..n}\varphi_i(t_i) \\ &= \sum_{\varphi_1,\ldots,\varphi_n\in\mathbb{S}^{\Sigma^*\emptyset}} \sum_{\substack{x_1\in {s^\flat}^{-1}(\varphi_1)\\ \cdots \\ x_n\in {s^\flat}^{-1}(\varphi_n)}} \psi(\sigma(x_1,\ldots,x_n))\cdot\prod_{i=1..n}\varphi_i(t_i) \\ &= \sum_{x_1,\ldots,x_n\in X} \psi(\sigma(x_1,\ldots,x_n))\cdot\prod_{i=1..n} s^\flat(x_i)(t_i)\end{aligned}$$

It follows that the diagram (13) commutes if and only if for all $\sigma(t_1,\ldots t_n)$ and all $x\in X$:

$$s^\flat(x)(\sigma(t_1,\ldots,t_n)) = \sum_{x_1,\ldots,x_n\in X} f(x)(\sigma(x_1,\ldots,x_n))\cdot\prod_{i=1..n} s^\flat(x_i)(t_i)$$

which is the semantics presented in Example 6.

The map $s^\flat$ is computed by induction; this is done by turning a weighted tree automaton f into the Σ-algebra $\hat{F}(f)$, which can be viewed as a *deterministic weighted bottom-up tree automaton*. We spell out the details. Given a coalgebra $f\colon X\to\mathcal{M}\Sigma X$ the computed Σ-algebra looks as follows:

$$\Sigma(\mathbb{S}^X) \xrightarrow{\rho_X} \mathbb{S}^{\Sigma X} \xrightarrow{\alpha_{\Sigma X}} \mathbb{S}^{\mathcal{M}\Sigma X} \xrightarrow{\mathbb{S}^f} \mathbb{S}^X$$

We have

$$\alpha_X(\varphi)(\psi) = \sum_{x\in X}\varphi(x)\cdot\psi(x)$$

and $\rho = \rho^\flat$:

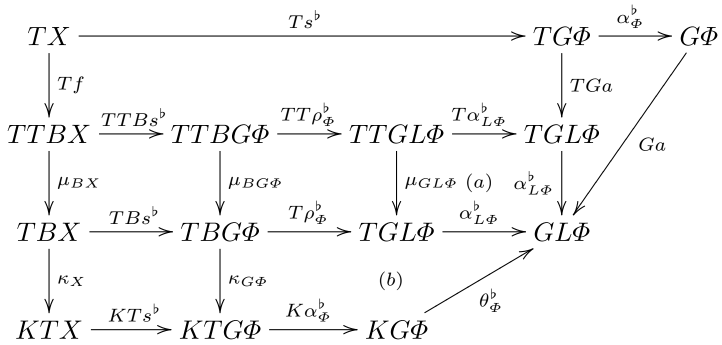
$$
\begin{aligned}
&(\mathbb{S}^{\Sigma\eta_X} \circ \mathbb{S}^{\rho^\flat_{\mathbb{S}X}} \circ \eta_{\Sigma\mathbb{S}X}(\sigma(\varphi_1,\ldots,\varphi_n)))(\tau(x_1,\ldots,x_m)) \\
&= (\mathbb{S}^{\rho^\flat_{\mathbb{S}X}\circ\Sigma\eta_X} \circ \lambda\psi.\psi(\sigma(\varphi_1,\ldots,\varphi_n)))(\tau(x_1,\ldots,x_m)) \\
&= (\rho^\flat_{\mathbb{S}X} \circ \Sigma\eta_X(\tau(x_1,\ldots,x_m)))(\sigma(\varphi_1,\ldots,\varphi_n)) \\
&= \rho^\flat_{\mathbb{S}X}(\tau(\lambda\varphi.\varphi(x_1),\ldots,\lambda\varphi.\varphi(x_m)))(\sigma(\varphi_1,\ldots,\varphi_n)) \\
&= \begin{cases} \prod_{i=1..n}\varphi_i(x_i) & \text{if } \sigma = \tau \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

The algebra is as follows:

$$
\begin{aligned}
\hat{F}(X,f)(\sigma(\varphi_1,\ldots,\varphi_n))(x) &= (\mathbb{S}^f \circ \alpha_{\Sigma X} \circ \rho_X(\sigma(\varphi_1,\ldots,\varphi_n)))(x) \\
&= (\alpha_{\Sigma X} \circ \rho_X(\sigma(\varphi_1,\ldots,\varphi_n)))(f(x)) \\
&= \sum_{t\in\Sigma X} f(x)(t)\cdot\rho_X(\sigma(\varphi_1,\ldots,\varphi_n))(t) \\
&= \sum_{x_1,\ldots,x_n\in X} f(x)(\sigma(x_1,\ldots,x_n))\cdot\prod_{i=1..n}\varphi_i(x_i)
\end{aligned}
$$

C Proof of Theorem 1

Proof. For TB-coalgebras, consider the following diagram.



where $a\colon L\Phi \to \Phi$ is the initial L-algebra and the rest is from the statement of the theorem. Everything commutes: (a) since $\alpha^\flat$ is an action, (b) by assumption, the upper left rectangle since $s^\flat$ is the logical semantics, and the rest by naturality. Commutativity of the outside means that $\alpha^\flat_\Phi \circ Ts^\flat$ is the (unique) logical semantics of θ on $T^\kappa(X,f)$. Now $\alpha^\flat_\Phi \circ Ts^\flat \circ \eta_X = \alpha^\flat_\Phi \circ \eta_{G\Phi} \circ s^\flat = s^\flat$ by naturality and the fact that $\alpha^\flat$ is an action.

For BT-coalgebras, we have the following diagram.

$$
\begin{array}{ccccccccc}
TX & & \xrightarrow{\quad Ts^\flat \quad} & & & & TG\Phi & \xrightarrow{\alpha^\flat_\Phi} & G\Phi \\
\downarrow{\scriptstyle Tf} & & & & & & \downarrow{\scriptstyle TGa} & & \downarrow{\scriptstyle Ga} \\
TBTX & \xrightarrow{TBTs^\flat} & TBTG\Phi & \xrightarrow{TB\alpha^\flat_\Phi} & TBG\Phi & \xrightarrow{T\rho^\flat_\Phi} & TGL\Phi & \xrightarrow{\alpha^\flat_{L\Phi}} & GL\Phi \\
\downarrow{\scriptstyle \kappa_{TX}} & & \downarrow{\scriptstyle \kappa_{TG\Phi}} & & \downarrow{\scriptstyle \kappa_{G\Phi}} & & (b) & \nearrow{\scriptstyle \theta^\flat_\Phi} & \\
KTTX & \xrightarrow{KTTs^\flat} & KTTG\Phi & \xrightarrow{KT\alpha^\flat_\Phi} & KTG\Phi & \xrightarrow{K\alpha^\flat_\Phi} & KG\Phi & & \\
\downarrow{\scriptstyle K\mu_X} & & \downarrow{\scriptstyle K\mu_{G\Phi}}\ (a) & & & \nearrow{\scriptstyle K\alpha^\flat_\Phi} & & & \\
KTX & \xrightarrow{KTs^\flat} & KTG\Phi & & & & & &
\end{array}
$$

where commutativity of (a) and (b) is as above, the rectangle commutes since $s^\flat$ is the logical semantics, and the rest commutes by naturality. Thus $\alpha^\flat_\Phi \circ Ts^\flat$ is the logical semantics of θ on $T_\kappa(X, f)$, and precomposition with η_X again yields the desired result.

D Proof of Corollary 1

Proof. Let $\bar{T}$ be either T^κ or T_κ, let (X, f) be a TB-coalgebra or a BT-coalgebra respectively, and $s^\flat$ the semantics of the forgetful logic on f. Under the above assumptions, by Theorem 1 we have $s^\flat = s^\flat_\theta \circ \eta_X$, where $s^\flat_\theta$ is the logical semantics of θ on $\bar{T}(X, f)$. Since $s^\flat_\theta$ is a logical semantics it factors through any coalgebra homomorphism, yielding condition (1) of correctness, and since it is expressive it decomposes as a coalgebra homomorphism followed by a mono, yielding condition (2).

E Proof of Theorem 2

Proof. Recall that $s^\flat$ is computed by transposing the unique morphism s from the initial L-algebra to $\tilde{F}(X, f)$, which means that $s^\flat = Gs \circ \iota_X$. Applying $\hat{G}$ to the initial algebra yields a final coalgebra, and therefore Gs is the morphism from $\hat{G}\tilde{F}(X, f)$ into the final coalgebra.

F Details of examples in Section 6

Example 7. The condition from Theorem 1 is commutativity of the following diagram:

$$
\begin{array}{ccccc}
\mathcal{M}(A \times \mathbb{S}^- + 1) & \xRightarrow{\mathcal{M}\rho^\flat} & \mathcal{M}(\mathbb{S}^{A\times - +1}) & \xRightarrow{\alpha^\flat_{A\times - +1}} & \mathbb{S}^{A\times - +1} \\
\Downarrow{\scriptstyle \kappa_{\mathbb{S}^-}} & & & & \| \\
\mathbb{S} \times (\mathcal{M}\mathbb{S}^-)^A & \xRightarrow{\mathsf{id}\times(\alpha^\flat)^A} & \mathbb{S} \times (\mathbb{S}^-)^A & \xRightarrow{\theta^\flat} & \mathbb{S}^{A\times - +1}
\end{array}
$$

Indeed, we have

$$K\alpha^\flat_\Phi \circ \kappa_{\mathbb{S}^\Phi}(\varphi) = (\varphi(*), \lambda a.\alpha^\flat(\lambda\psi.\varphi(a,\psi))) = (\varphi(*), \lambda a.\lambda w. \sum_{\psi\in\mathbb{S}^\Phi} \varphi(a,\psi)\cdot\psi(w))$$

and thus

$$\begin{aligned}(\theta^\flat_\Phi \circ K\alpha^\flat_\Phi \circ \kappa_{\mathbb{S}^\Phi}(\varphi))(*) &= \varphi(*)\\ (\theta^\flat_\Phi \circ K\alpha^\flat_\Phi \circ \kappa_{\mathbb{S}^\Phi}(\varphi))(a,w) &= \sum_{\psi\in\mathbb{S}^\Phi} \varphi(a,\psi)\cdot\psi(w)\end{aligned}$$

which coincides with $\alpha^\flat_{A\times\Phi+1} \circ \mathcal{M}\delta^\flat_\Phi$ as computed (in a more general setting) in Appendix B.

Example 8. We treat the determinization $\langle\tau^o,\tau^t\rangle$ described in the example. The relevant condition of Theorem 1 instantiates to commutativity of:

$$\begin{array}{ccccc}
\mathcal{P}_\omega(2\times(2^-)^A) & \xRightarrow{\mathcal{P}_\omega\rho^\flat} & \mathcal{P}_\omega(2^{A\times-+1}) & \xRightarrow{\beta^\flat L} & 2^{A\times-+1}\\
\Downarrow{\scriptstyle\langle\tau^o_{2^-},\tau^t_{2^-}\rangle} & & & & \|\\
2\times(\mathcal{P}_\omega(2^-))^A & \xRightarrow{\mathsf{id}\times(\beta^\flat)^A} & 2\times(2^-)^A & \xRightarrow{\rho^\flat} & 2^{A\times-+1}
\end{array}$$

We have, for any set Φ:

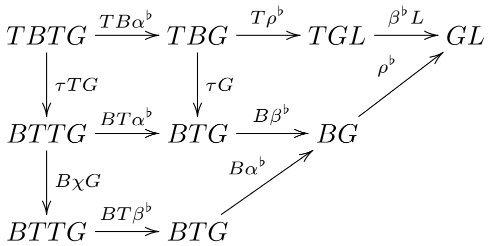
$$\begin{aligned}(\beta^\flat_{L\Phi}\circ\mathcal{P}_\omega\rho^\flat_\Phi)(S)(*) = \mathsf{tt} &\text{ iff } \forall\varphi\in(\mathcal{P}_\omega\rho^\flat_\Phi)(S).\varphi(*)=\mathsf{tt}\\
&\text{ iff } \forall(o,t)\in S.o=\mathsf{tt}\\
&\text{ iff } \tau^o_{2^\Phi}(S)=\mathsf{tt}\\
&\text{ iff } (\rho^\flat_\Phi\circ B\beta^\flat_\Phi\circ\langle\tau^o_{2^\Phi},\tau^t_{2^\Phi}\rangle)(S)(*)=\mathsf{tt}\end{aligned}$$

and for any $a\in A$, $w\in\Phi$:

$$\begin{aligned}(\beta^\flat_{L\Phi}\circ\mathcal{P}_\omega\rho^\flat_\Phi)(S)(a,w) = \mathsf{tt} &\text{ iff } \forall\varphi\in(\mathcal{P}_\omega\rho^\flat_\Phi)(S).\varphi(a,w)=\mathsf{tt}\\
&\text{ iff } \forall(o,t)\in S.t(a)(w)=\mathsf{tt}\\
&\text{ iff } \forall\varphi\in\tau^t_{2^\Phi}(S)(a).\varphi(w)=\mathsf{tt}\\
&\text{ iff } \beta^\flat_\Phi(\tau^t_{2^\Phi}(S)(a))(w)=\mathsf{tt}\\
&\text{ iff } ((\beta^\flat_\Phi)^A\circ\tau^t_{2^\Phi})(S)(a)(w)=\mathsf{tt}\\
&\text{ iff } (\rho^\flat_\Phi\circ B\beta^\flat_\Phi\circ\langle\tau^o_{2^\Phi},\tau^t_{2^\Phi}\rangle)(S)(a,w)=\mathsf{tt}\end{aligned}$$

which proves commutativity of the diagram.

Example 9. The condition of Theorem 1 in this case is that the following commutes:

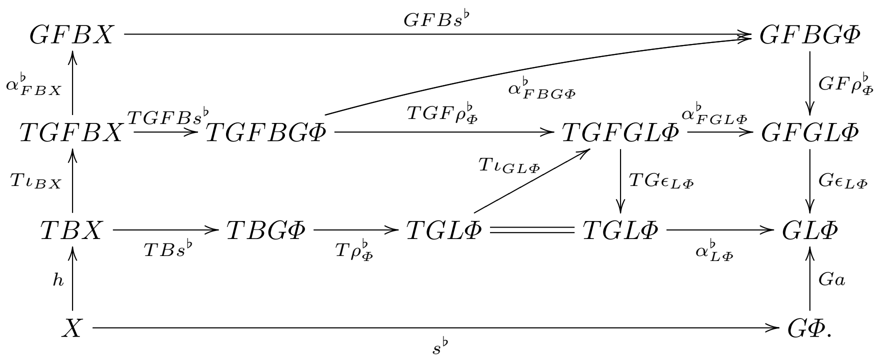


The square commutes by naturality, and the upper right shape is diagram for proving correctness of the determinization procedure τ, considered in Example 8 (commutativity is proved above). The lower shape expresses that χ distributes conjunction over disjunction, which is the case indeed.

G Details of Section 7.1

Lemma 2. *For any $h\colon X \to TBX$, the semantics of $\bar{\alpha} \circledcirc \rho$ on $\Gamma(h) = \gamma_{BX} \circ h$ coincides with the semantics of $\alpha \circledcirc \rho$ on h.*

Proof. For any coalgebra $h\colon X \to TBX$, consider the diagram:



Here the top row commutes by naturality of $\alpha^\flat$, and the middle row by naturality of ι, by (11), and by naturality of $\alpha^\flat$. As a result, the outer shape commutes if and only if the bottom row does. The bottom row defines $s^\flat$ as the logical semantics of $\alpha \circledcirc \rho$ on h (see (3) in Section 3). Similarly, the outer shape defines $s^\flat$ as the logical semantics of $\bar{\alpha} \circledcirc \rho$ on $\gamma_{BX} \circ h$. Since both diagrams define $s^\flat$ uniquely, the two logical semantics must coincide. □

Lemma 3. *The determinization procedure $(GF)^\kappa$ defined as in (6) is correct with respect to $\bar{\alpha} \circledcirc \rho$.*

Proof. We use Corollary 1, where we put $T = GF$, $\alpha = \bar{\alpha}$ defined above, $K = B$, and $\theta = \rho$. Obviously then θ is expressive, and it is easy to check that $\bar{\alpha} = G\epsilon$ is an action. The only remaining condition is the diagram from Theorem 1, which is the outer shape of:

$$
\begin{array}{ccccc}
GFBG & \xRightarrow{GF\rho^\flat} & GFGL & & \\
{\scriptstyle G\rho G}\Downarrow & & \Downarrow{\scriptstyle G\epsilon L} & & \\
GLFG & \xRightarrow{GL\epsilon} & GL & & \\
{\scriptstyle (\rho^\flat)^{-1}FG}\Downarrow & & \Downarrow{\scriptstyle (\rho^\flat)^{-1}} & \overset{\mathsf{id}}{\searrow\!\!\searrow} & \\
BGFG & \xRightarrow[BG\epsilon]{} & BG & \xRightarrow[\rho^\flat]{} & GL.
\end{array}
$$

Here, the top square commutes by Lemma 1 and the bottom square by naturality of $(\rho^\flat)^{-1}$.

H Proof of Theorem 3

Consider any coalgebra $f\colon X \to TBX$. The B-coalgebra $\hat{G}\tilde{F}(X, f)$ is defined as in (8). Recall that $\Gamma(X, f) = \gamma_{BX} \circ f$, for γ defined by (9). Recall also that $(GF)^\kappa$ is defined as in (6), for κ as in (10). Combining all this, $(GF)^\kappa \circ \Gamma(X, f)$ is the coalgebra:

$$
\begin{array}{ccccccccc}
GFX & \xrightarrow{GFf} & GFTBX & & & & GFBX & \xrightarrow{G\rho_X} & GLFX \xrightarrow{(\rho^\flat)^{-1}_{FX}} BGFX. \\
& & {\scriptstyle GFT\iota_{BX}}\downarrow & & & & \uparrow{\scriptstyle G\epsilon_{FBX}} & & \\
& & GFTGFBX & & \xrightarrow{GF\alpha^\flat_{FBX}} & & GFGFBX & &
\end{array}
$$

The first and the last two components of it are identical to those of (8). To show the remaining components equal, instantiate the following diagram at BX and mapped along G:

$$
\begin{array}{ccc}
F & \xRightarrow{\alpha} & FT \\
{\scriptstyle \epsilon F}\Downarrow & & \Uparrow{\scriptstyle FT\iota} \\
FGF & \xRightarrow[F\alpha^\flat F]{} & FTGF.
\end{array}
$$

It is easy to check that this commutes, from the definition of $\alpha^\flat$ from α as in (2).