Trace Semantics via Determinization[☆]

Bart Jacobs[a], Alexandra Silva[a,1,*], Ana Sokolova[b]

[a] *Institute for Computing and Information Sciences, Radboud University Nijmegen*
[b] *Department of Computer Sciences, University of Salzburg*

Abstract

This paper takes a fresh look at the topic of trace semantics in the theory of coalgebras. In the last few years, two approaches, somewhat incomparable at first sight, captured successfully in a coalgebraic setting trace semantics for various types of transition systems. The first development of coalgebraic trace semantics used final coalgebras in Kleisli categories and required some non-trivial assumptions, which do not always hold, even in cases where one can reasonably speak of traces (like for weighted automata). The second development stemmed from the observation that trace semantics can also arise by performing a determinization construction and used final coalgebras in Eilenberg-Moore categories. In this paper, we develop a systematic study in which the two approaches can be studied and compared. Notably, we show that the two different views on trace semantics are equivalent, in the examples where both approaches are applicable.

Keywords: coalgebra, Kleisli category, Eilenberg-Moore category, trace semantics

1. Introduction

Studying the semantics of state-based systems has been a long quest in Theoretical Computer Science. Central to this study is the search for the correct notion of equivalence and proof methods thereof. Coalgebras provide an abstract framework for state-based computation, where a canonical notion of equivalence can be uniformly derived from the type (formally, a functor) of the system under study. The canonical notion of equivalence is adequate in many situations, but turns out to be too fine when the type of the system includes certain computational effects that are intended to be hidden from the observer. To overcome this shortcoming of the general theory of coalgebras, notions of equivalence in which the type of the system can be split into relevant type information and computational effects to be hidden were devised. The coarser equivalences were coined under the generic name of *coalgebraic (decorated) trace semantics* [18, 36, 37, 6].

In this paper, we take a fresh look at the topic of coalgebraic trace semantics and provide a framework where the existing two main approaches, which we explain next, can be studied and compared. Throughout the rest of the paper we assume a certain familiarity with basic notions of category theory, such as functors, natural transformations and distributive laws, as well as with the definitions of Kleisli and Eilenberg-Moore categories. Readers unfamiliar with these notions can find basic material on these in standard category theory books, see e.g. [2].

A coalgebra is a map of the form $X \to H(X)$, where X is a state space and H is a functor that captures the kind of computation involved. Often, this H is, or contains, a monad T, providing certain computational effects, such as when T is lift $1 + (-)$, powerset $\mathcal{P}$ or distribution $\mathcal{D}$, giving partial, non-deterministic or probabilistic computation.

[☆] Extended version of [23].

[*] Corresponding author

Email addresses: bart@cs.ru.nl (Bart Jacobs), alexandra@cs.ru.nl (Alexandra Silva), anas@cs.uni-salzburg.at (Ana Sokolova)

[1] Also affiliated to Centrum Wiskunde & Informatica (Amsterdam, The Netherlands) and HASLab / INESC TEC, Universidade do Minho (Braga, Portugal).

In the case such an H contains a monad T, it turns out that there are two archetypical forms in which the monad T plays a role, namely in:

$$X \longrightarrow G(TX) \qquad \text{or} \qquad X \longrightarrow T(FX) \tag{1}$$

In the first case, the monad T occurs inside a functor G that typically handles input and output. In the second case the monad T is on the outside, encapsulating a form of computation over a functor F, that typically describes the transitions involved. In some cases these two forms are equivalent, for instance for non-deterministic automata with labels—given by a set A—and termination. They involve the powerset monad $\mathcal{P}$ and can be described equivalently as:

$$X \longrightarrow 2 \times (\mathcal{P}X)^A \qquad \text{or} \qquad X \longrightarrow \mathcal{P}(1 + A \times X)$$

where on the left-hand-side we have the functor $G = 2 \times (-)^A$ and on the right-hand-side $F = 1 + A \times (-)$. These descriptions are equivalent because the powerset monad $\mathcal{P}$ is special (it is "additive", see [12], mapping coproducts to products). In fact, there are isomorphisms:

$$\mathcal{P}(1 + A \times X) \;\cong\; \mathcal{P}(1) \times \mathcal{P}(A \times X) \;\cong\; 2 \times (\mathcal{P}(X))^A. \tag{2}$$

Classically, to study language or trace equivalence for non-deterministic automata there is a standard construction called determinization [20]. It involves changing the state space X into a state space $\mathcal{P}(X)$. In doing so, the transition structure becomes much simpler (in particular, deterministic).

In this paper, we are interested in a similar determinization construction, but on a more abstract level. It involves changing the state space from X into $T(X)$, for a monad T. It turns out that this can be done relatively easily for coalgebras of the form $X \to G(TX)$, on the left in (1), as illustrated in [36]: it involves a distributive law between G and T, and such law corresponds to a lifting $\widehat{G}$ of the functor G to the category $\mathcal{EM}(T)$ of Eilenberg-Moore algebras of T. Moreover, the final G-coalgebra lifts to a final $\widehat{G}$-coalgebra in $\mathcal{EM}(T)$. In this way, one obtains final coalgebra semantics in a category of algebras. It yields a first form of "$\mathcal{EM}$" extension semantics, see Definition 1. Categorically, this extension $X \mapsto T(X)$ is given by the free algebra functor $\mathbf{C} \to \mathcal{EM}(T)$.

Extension for coalgebras of the form $X \to T(FX)$ on the right in (1) is more complicated; it involves the comparison functor $\mathcal{K}\ell(T) \to \mathcal{EM}(T)$. We proceed by first translating these coalgebras to coalgebras of the lifted functor $\widehat{G}$ via a suitable law $\mathfrak{e}\colon TF \Rightarrow GT$, see Section 6 (and [3]). It will be shown that the resulting trace semantics, in categories of algebras, as sketched above,

- not only includes the trace semantics in Kleisli categories developed in [18];
- but also covers more examples; in particular, it covers trace semantics for weighted automata $X \to \mathcal{M}(1 + A \times X)$, involving the multiset monad $\mathcal{M}$. This monad does not fit in the trace framework of [18] because its Kleisli category is not dcpo-enriched.

The technical (categorical) core of the paper concentrates on lifting the comparison functor $\mathcal{K}\ell(T) \to \mathcal{EM}(T)$ to categories of coalgebras $\mathbf{CoAlg}(\widehat{F}) \to \mathbf{CoAlg}(\widehat{G})$, of lifted functors $\widehat{F}, \widehat{G}$. We specialize this framework by taking G to be the functor $B \times (-)^A$ for deterministic automata. Its final coalgebra $B^{A^\star}$ gives trace semantics. Our framework is general enough to allow a different semantics, like "tree" semantics, by using a different functor G, as we illustrate in other examples.

The paper is organized as follows. The first section below recalls the basics about monads and the associated Kleisli category and category of (Eilenberg-Moore) algebras. Section 3 gives a systematic account of liftings of functors to such (Kleisli and algebra) categories, and of distributive laws; it includes a lifting result for final coalgebras to categories of Eilenberg-Moore algebras. Subsequently, Section 4 briefly recalls the coalgebraic description of deterministic automata, their final coalgebras, and the lifting of these final coalgebras to categories of Eilenberg-Moore algebras; it also includes a description of such final coalgebras obtained via a lifting of an adjunction $\mathbf{Sets}^{\mathrm{op}} \leftrightarrows \mathcal{EM}(T)$, in the style of [34]. Section 5 contains examples of the application of $\mathcal{EM}$-extension semantics: for classical non-deterministic automata and a new example mixing probability and non-determinism, for simple Segala systems, applicable also to alternating automata and non-deterministic weighted automata, as well as to pushdown automata. In Section 6, we develop an extension semantics for coalgebras of type TF (Definition 2, via Theorem 2) and show

that the Kleisli trace semantics from [18] fits in the current setting (Proposition 5). We also summarize and relate all relevant categories and functors (Theorem 3). Section 7 presents examples of the $\mathcal{K}\ell$-extension semantics, including examples already treated in [18] and, more notably, examples that cannot be studied in the framework of [18]. Section 8 contains concluding remarks and pointers for future work.

This paper is an extended version of [23]. We include a new section on duality and extra examples: alternating, non-deterministic weighted, and pushdown automata.

2. Monads and their Kleisli and Eilenberg-Moore categories

This section recalls the basics of the theory of monads, as needed here. For more information, see e.g. [29, 4, 28, 7]. A monad is a functor $T\colon \mathbf{C} \to \mathbf{C}$ together with two natural transformations: a unit $\eta\colon \mathrm{id}_{\mathbf{C}} \Rightarrow T$ and multiplication $\mu\colon T^2 \Rightarrow T$. These are required to make the following diagrams commute, for $X \in \mathbf{C}$.

$$\begin{array}{ccccc} T(X) & \xrightarrow{\eta_{T(X)}} & T^2(X) & \xleftarrow{T(\eta_X)} & T(X) \\ & \searrow & \downarrow{\scriptstyle \mu_X} & \swarrow & \\ & & T(X) & & \end{array} \qquad \begin{array}{ccc} T^3(X) & \xrightarrow{\mu_{T(X)}} & T^2(X) \\ {\scriptstyle T(\mu_X)}\downarrow & & \downarrow{\scriptstyle \mu_X} \\ T^2(X) & \xrightarrow[\mu_X]{} & T(X) \end{array}$$

We briefly describe the examples of monads on $\mathbf{Sets}$ that we use in this paper.

- The powerset monad $\mathcal{P}$ maps a set X to the set $\mathcal{P}X$ of subsets of X, and a function $f\colon X \to Y$ to $\mathcal{P}(f)\colon \mathcal{P}X \to \mathcal{P}Y$ given by direct image. Its unit is given by singleton $\eta(x) = \{x\}$ and multiplication by union $\mu(\{X_i \in \mathcal{P}X \mid i \in I\}) = \bigcup_{i\in I} X_i$.

- The subprobability distribution monad $\mathcal{D}$ is defined, for a set X and a function $f\colon X \to Y$, as

$$\mathcal{D}X \;=\; \{\varphi\colon X \to [0,1] \mid \textstyle\sum_{x\in X} \varphi(x) \leq 1\} \qquad \mathcal{D}f(\varphi)(y) \;=\; \sum_{x\in f^{-1}(y)} \varphi(x).$$

  The support set of a distribution $\varphi \in \mathcal{D}X$ is defined as

$$\mathrm{supp}(\varphi) = \{x \in X \mid \varphi(x) \neq 0\}.$$

  Note that we do not restrict distributions to finitely supported ones in the definition of $\mathcal{D}$. The unit of $\mathcal{D}$ is given by a Dirac distribution $\eta(x) = \delta_x = (x \mapsto 1)$ for $x \in X$ and the multiplication by $\mu(\Phi)(x) = \sum_{\varphi\in\mathrm{supp}(\Phi)} \Phi(\varphi)\cdot\varphi(x)$ for $\Phi \in \mathcal{D}\mathcal{D}X$.

- For a semiring S, the multiset monad $\mathcal{M}_S$ is defined on a set X as:

$$\mathcal{M}_S X \;=\; \{\varphi\colon X \to S \mid \mathrm{supp}(\varphi) \text{ is finite }\}.$$

  This monad captures multisets $\varphi \in \mathcal{M}_S X$, where the value $\varphi(x) \in S$ gives the multiplicity of the element $x \in X$. When $S = \mathbb{N}$, this is sometimes called the bag monad.

  On functions, $\mathcal{M}_S$ is defined like the subdistribution monad $\mathcal{D}$. Again, the support set of a multiset $\varphi \in \mathcal{M}_S X$ is defined as $\mathrm{supp}(\varphi) = \{x \in X \mid \varphi(x) \neq 0\}$. The finite support requirement is needed for $\mathcal{M}$ to be a monad. The unit η and multiplication μ of $\mathcal{M}_S$ are defined in the same way as for $\mathcal{D}$.

We also use the non-deterministic side-effect monad whose definition we postpone to Section 5.5.

With a monad T on a category $\mathbf{C}$ one associates two categories and a comparison functor E between them, as on the right. The "Kleisli" category $\mathcal{K}\ell(T)$ is used to capture computations of type T, in the paradigm that uses monads for effects in a functional world [31]. The objects of the "Eilenberg-Moore" category $\mathcal{EM}(T)$ are algebraic structures, in abstract form. Objects of $\mathcal{EM}(T)$ are called 'algebras', or sometimes 'Eilenberg-Moore algebras' to avoid possible confusion with algebras of a functor F. The latter also form a category written as $\mathbf{Alg}(F)$. The comparison functor $E\colon \mathcal{K}\ell(T) \to \mathcal{EM}(T)$ plays here the role of pure determinization operation. The categories $\mathcal{K}\ell(T)$ and $\mathcal{EM}(T)$ are initial and final in a suitable sense, see [29] for details. This comparison functor E will be used as the determinization functor. We now describe the above points in more detail.

$$\begin{array}{ccc} \mathcal{K}\ell(T) & \xrightarrow{\ E\ } & \mathcal{EM}(T) \\ & \searrow \quad \swarrow & \\ & \mathbf{C} & \end{array}$$

The Kleisli category $\mathcal{K}\ell(T)$ has the same objects as the underlying category $\mathbf{C}$, but morphisms $X \to Y$ in $\mathcal{K}\ell(T)$ are maps $X \to T(Y)$ in $\mathbf{C}$. The identity map $X \to X$ in $\mathcal{K}\ell(T)$ is T's unit $\eta_X\colon X \to T(X)$; and composition $g \odot f$ in $\mathcal{K}\ell(T)$ uses T's multiplication in: $g \odot f = \mu \circ T(g) \circ f$. There is a forgetful functor $\mathcal{U}\colon \mathcal{K}\ell(T) \to \mathbf{C}$, sending X to $T(X)$ and f to $\mu \circ T(f)$. This functor has a left adjoint $\mathcal{F}$, given by $\mathcal{F}(X) = X$ and $\mathcal{F}(f) = \eta \circ f$. Such a Kleisli category $\mathcal{K}\ell(T)$ inherits colimits from the underlying category $\mathbf{C}$.

The category $\mathcal{EM}(T)$ of Eilenberg-Moore algebras has as objects maps of the form $a\colon T(X) \to X$, making the first two diagrams below commute.

$$\begin{array}{ccc} X & \xrightarrow{\ \eta\ } & TX \\ & \searrow^{=} & \downarrow{\scriptstyle a} \\ & & X \end{array} \qquad \begin{array}{ccc} T^2X & \xrightarrow{\ T(a)\ } & TX \\ {\scriptstyle \mu}\downarrow & & \downarrow{\scriptstyle a} \\ TX & \xrightarrow[\ a\ ]{} & X \end{array} \qquad \begin{array}{ccc} TX & \xrightarrow{\ T(f)\ } & TY \\ {\scriptstyle a}\downarrow & & \downarrow{\scriptstyle b} \\ X & \xrightarrow[\ f\ ]{} & Y \end{array}$$

A homomorphism of algebras $\big(TX \xrightarrow{a} X\big) \to \big(TY \xrightarrow{b} Y\big)$ is a map $f\colon X \to Y$ in $\mathbf{C}$ between the underlying objects making the diagram above on the right commute. The diagram in the middle thus says that the map a is a homomorphism $\mu \to a$. The forgetful functor $\mathcal{U}\colon \mathcal{EM}(T) \to \mathbf{C}$ has a left adjoint $\mathcal{F}$, mapping an object $X \in \mathbf{X}$ to the (free) algebra $\mu_X\colon T^2(X) \to T(X)$ with carrier $T(X)$.

Each category $\mathcal{EM}(T)$ inherits limits from the category $\mathbf{C}$. In the special case where $\mathbf{C} = \mathbf{Sets}$, the category of sets and functions (our standard universe), the category $\mathcal{EM}(T)$ is not only complete but also cocomplete (see [4, § 9.3, Prop. 4]).

The extension functor $E\colon \mathcal{K}\ell(T) \to \mathcal{EM}(T)$ sends an object $X \in \mathcal{K}\ell(T)$ to the free algebra $E(X) = (\mu\colon T^2(X) \to T(X))$. For a morphism $f\colon X \to Y$ in $\mathcal{K}\ell(T)$, that is, $f\colon X \to T(Y)$ in $\mathbf{C}$, we have $E(f) = \mu \circ T(f)\colon T(X) \to T(Y)$. It forms a map of algebras. Sometimes this $E(f)$ is called the "Kleisli extension" of f.

3. Liftings to Kleisli and Eilenberg-Moore categories

In this section we consider the situation where we have a monad $T\colon \mathbf{C} \to \mathbf{C}$, with unit η and multiplication μ, and two endofunctors $F, G\colon \mathbf{C} \to \mathbf{C}$ on the same category $\mathbf{C}$. We will be interested in distributive laws between T, and F or G. These can be of two forms, namely:

$$\begin{array}{lll} FT \Longrightarrow TF & \text{"}F \text{ distributes over } T\text{"} & \text{"}\mathcal{K}\ell\text{-law"} \\ TG \Longrightarrow GT & \text{"}T \text{ distributes over } G\text{"} & \text{"}\mathcal{EM}\text{-law"} \end{array}$$

It is rather difficult to remember whether the monad distributes over a functor or vice versa and so we prefer to use the terminology "$\mathcal{K}\ell$-law" and "$\mathcal{EM}$-law". This is justified by Proposition 1 below.

But first we have to be more precise about what a distributive law is. A $\mathcal{K}\ell$-law $\lambda\colon FT \Rightarrow TF$ is a natural transformation that commutes appropriately with the unit and multiplication of the monad, i.e., for all X in $\mathbf{C}$ the following diagrams commute:

$$\begin{array}{ccc} FX & = & FX \\ {\scriptstyle F(\eta_X)}\downarrow & & \downarrow{\scriptstyle \eta_{FX}} \\ FTX & \xrightarrow[\ \lambda_X\ ]{} & TFX \end{array} \qquad \begin{array}{ccccc} FT^2X & \xrightarrow{\ \lambda_{TX}\ } & TFTX & \xrightarrow{\ T(\lambda_X)\ } & T^2FX \\ {\scriptstyle F(\mu_X)}\downarrow & & & & \downarrow{\scriptstyle \mu_{FX}} \\ FTX & & \xrightarrow[\qquad\lambda_X\qquad]{} & & TFX \end{array} \tag{3}$$

An $\mathcal{EM}$-law $\rho\colon TG \Rightarrow GT$ is a natural transformation for which the following diagrams commute for all X in $\mathbf{C}$.

$$\begin{array}{ccc} GX & = & GX \\ {\scriptstyle \eta_{GX}}\downarrow & & \downarrow{\scriptstyle G(\eta_X)} \\ TGX & \xrightarrow[\rho_X]{} & GTX \end{array} \qquad\qquad \begin{array}{ccccc} T^2GX & \xrightarrow{T(\rho_X)} & TGTX & \xrightarrow{\rho_{TX}} & GT^2X \\ {\scriptstyle \mu_{GX}}\downarrow & & & & \downarrow{\scriptstyle G(\mu_X)} \\ TGX & & \xrightarrow[\rho_X]{} & & GTX \end{array} \tag{4}$$

The following "folklore" result gives an alternative description of distributive laws in terms of liftings to Kleisli and Eilenberg-Moore categories, see also [25], [32] or [3].

Proposition 1 ("laws and liftings"). *Assume a monad T and endofunctors F, G on the same category $\mathbf{C}$, as above. There are bijective correspondences between $\mathcal{K}\ell$ ($\mathcal{EM}$)-laws and liftings of F (G) to $\mathcal{K}\ell$ ($\mathcal{EM}$)-categories, in:*

$$\begin{array}{c} \mathcal{K}\ell\text{-law } \lambda\colon FT \Rightarrow TF \\ \hline\hline \mathcal{K}\ell(T) \xrightarrow{L} \mathcal{K}\ell(T) \\ \downarrow \qquad\quad \downarrow \\ \mathbf{C} \xrightarrow{F} \mathbf{C} \end{array} \qquad\qquad \begin{array}{c} \mathcal{EM}\text{-law } \rho\colon TG \Rightarrow GT \\ \hline\hline \mathcal{EM}(T) \xrightarrow{R} \mathcal{EM}(T) \\ \downarrow \qquad\quad \downarrow \\ \mathbf{C} \xrightarrow{G} \mathbf{C} \end{array}$$

PROOF. Assuming a $\mathcal{K}\ell$-law $\lambda\colon FT \Rightarrow TF$ we can define $L\colon \mathcal{K}\ell(T) \to \mathcal{K}\ell(T)$ as:

$$L(X) \;=\; F(X) \qquad L\big(X \xrightarrow{f} Y\big) \;=\; \big(F(X) \xrightarrow{F(f)} F(TY) \xrightarrow{\lambda_X} T(FY)\big).$$

The above two requirements (3) for λ precisely say that L is a functor.

Conversely, assume there is a functor $L\colon \mathcal{K}\ell(T) \to \mathcal{K}\ell(T)$ in a commuting square as described in the proposition. Then, on objects, $L(X) = F(X)$. Further, for a map $f\colon X \to TY$ in $\mathbf{C}$ we get $L(f)\colon FX \to TFY$ in $\mathbf{C}$. This suggests how to define a distributive law: the identity map $\mathrm{id}_{TX}\colon TX \to TX$ in $\mathbf{C}$ forms a map $TX \to X$ in $\mathcal{K}\ell(T)$, so that we can define $\lambda_X = L(\mathrm{id}_{TX})\colon FTX \to TFX$ in $\mathbf{C}$. It satisfies (3).

For the second correspondence assume we have an $\mathcal{EM}$-law $\rho\colon TG \Rightarrow GT$. It gives rise to a functor $R\colon \mathcal{EM}(T) \to \mathcal{EM}(T)$ by:

$$\begin{pmatrix} TX \\ \downarrow a \\ X \end{pmatrix} \;\longmapsto\; \begin{pmatrix} TGX \\ \downarrow G(a)\circ\rho \\ GX \end{pmatrix} \qquad \text{and} \qquad f \;\longmapsto\; G(f).$$

The equations (4) guarantee that this yields a new T-algebra.

In the reverse direction, assume a lifting $R\colon \mathcal{EM}(T) \to \mathcal{EM}(T)$. Applying it to the multiplication μ_X yields an algebra $R(\mu_X)\colon T(GTX) \to GTX$. We then define $\rho_X = R(\mu_X) \circ TG(\eta_X)\colon TGX \to GTX$. Remaining details are left to the reader. □

In what follows we shall simply write $\widehat{F}$, $\widehat{G}$ for the lifting of F, G, both when it comes from a $\mathcal{K}\ell$-law λ or from an $\mathcal{EM}$-law ρ. Usually these laws are fixed, so confusion is unlikely, and a light, overloaded notation is preferred.

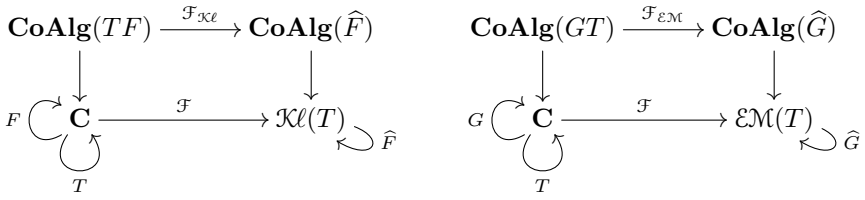
The next result (see also [3]) is not really used in this paper, but it is a natural sequel to the previous proposition since it relates the liftings $\widehat{F}, \widehat{G}$ to the standard adjunctions. Recall that we write $\mathbf{Alg}(-)$ and $\mathbf{CoAlg}(-)$ for categories of algebras and coalgebras of a *functor*, not of a (co)monad.

Proposition 2. *In presence of a $\mathcal{K}\ell$-law and an $\mathcal{EM}$-law, the adjunctions $\mathbf{C} \rightleftarrows \mathcal{K}\ell(T)$ and $\mathbf{C} \rightleftarrows \mathcal{EM}(T)$ lift to adjunctions between categories of, respectively, algebras and coalgebras, as described below.*

$$\begin{array}{ccc} \mathbf{Alg}(F) & \underset{\widehat{\mathcal{U}}}{\overset{\widehat{\mathcal{F}}}{\rightleftarrows}} & \mathbf{Alg}(\widehat{F}) \\ \downarrow & & \downarrow \\ \mathbf{C} & \underset{\mathcal{U}}{\overset{\mathcal{F}}{\rightleftarrows}} & \mathcal{K}\ell(T) \end{array} \qquad\qquad \begin{array}{ccc} \mathbf{CoAlg}(G) & \underset{\widehat{\mathcal{U}}}{\overset{\widehat{\mathcal{F}}}{\rightleftarrows}} & \mathbf{CoAlg}(\widehat{G}) \\ \downarrow & & \downarrow \\ \mathbf{C} & \underset{\mathcal{U}}{\overset{\mathcal{F}}{\rightleftarrows}} & \mathcal{EM}(T) \end{array}$$

There is another lifting result, for free functors only, that is relevant in this setting.

Lemma 1. *In presence of a $\mathcal{K}\ell$-law the free functor $\mathcal{F}\colon \mathbf{C} \to \mathcal{K}\ell(T)$ can be lifted, and similarly, given an $\mathcal{EM}$-law the free algebra functor $\mathcal{F}\colon \mathbf{C} \to \mathcal{EM}(T)$ can be lifted:*



(5)

The functor $\mathbf{CoAlg}(GT) \to \mathbf{CoAlg}(\widehat{G})$ on the right gives an abstract description of what is called the generalized powerset construction in [36].

PROOF. The first part is easy, since the functor $\mathcal{F}_{\mathcal{K}\ell}\colon \mathbf{CoAlg}(TF) \to \mathbf{CoAlg}(\widehat{F})$ is the identity on objects; it sends a map f of TF-coalgebras to $\mathcal{F}(f) = \eta \circ f$.

Next, assuming an $\mathcal{EM}$-law $\rho\colon TG \Rightarrow GT$ one defines $\mathcal{F}_{\mathcal{EM}}\colon \mathbf{CoAlg}(GT) \to \mathbf{CoAlg}(\widehat{G})$ by

$$\mathcal{F}_{\mathcal{EM}}\Big(X \xrightarrow{c} GTX\Big) \;=\; \Big(TX \xrightarrow{T(c)} TGTX \xrightarrow{\rho_{TX}} GT^2X \xrightarrow{G(\mu_X)} GTX\Big). \tag{6}$$

It is not hard to see that $\mathcal{F}_{\mathcal{EM}}(c)$ is a coalgebra $\mu_X \to \widehat{G}(\mu_X)$ on the free algebra μ_X. On morphisms one simply has $\mathcal{F}_{\mathcal{EM}}(f) = T(f)$. □

$\mathcal{K}\ell$-laws are used to obtain final coalgebras in Kleisli categories ([18]) but this requires non-trivial side-conditions, like enrichement in dcpo's. For $\mathcal{EM}$-laws the situation is much easier, see below; instances of this result have been studied in [36], see also [3].

Proposition 3. *Assume a monad T and endofunctor G on a category $\mathbf{C}$, with an $\mathcal{EM}$-law $\rho\colon TG \Rightarrow GT$ between them. If G has a final coalgebra $\zeta\colon Z \xrightarrow{\cong} GZ$ in $\mathbf{C}$, then Z carries a T-algebra structure obtained by finality, as on the left below. The map ζ then forms a map of algebras as on the right, which is the final coalgebra for the lifted functor $\widehat{G}\colon \mathcal{EM}(T) \to \mathcal{EM}(T)$.*

$$\begin{array}{ccc} GTZ & \xrightarrow{G(\alpha)} & GZ \\ {\scriptstyle \rho\circ T(\zeta)}\uparrow & & \uparrow{\scriptstyle \cong\ \zeta} \\ TZ & \xrightarrow[\alpha]{} & Z \end{array} \qquad \begin{pmatrix} TZ \\ \downarrow{\scriptstyle \alpha} \\ Z \end{pmatrix} \xrightarrow[\cong]{\zeta} \widehat{G}\begin{pmatrix} TZ \\ \downarrow{\scriptstyle \alpha} \\ Z \end{pmatrix} = \begin{pmatrix} T(GZ) \\ \downarrow{\scriptstyle G(\alpha)\circ\rho} \\ GZ \end{pmatrix}$$

PROOF. We leave it to the reader to verify that α is a T-algebra. By construction of α, the map ζ is an algebra homomorphism $\alpha \to \widehat{G}(\alpha)$. Suppose for an arbitrary algebra $b\colon TY \to Y$ we have a $\widehat{G}$-coalgebra $c\colon b \to \widehat{G}(b)$. Then $c\colon Y \to GY$ satisfies $G(b) \circ \rho \circ T(c) = c \circ b$. By finality in $\mathbf{C}$ there is a unique map $f\colon Y \to Z$ with $\zeta \circ f = G(f) \circ c$. This f is then the unique map $b \to \alpha$ in $\mathcal{EM}(T)$. □

At this stage we can describe the first form of extension semantics, which we will call $\mathcal{EM}$-extension semantics, since it depends on an $\mathcal{EM}$-law.

Definition 1 ($\mathcal{EM}$-extension). *Let $\rho\colon TG \Rightarrow GT$ be an $\mathcal{EM}$-law, for $G, T\colon \mathbf{C} \to \mathbf{C}$, such that the final G-coalgebra $Z \xrightarrow{\cong} GZ$ exists. "Extension" semantics $X \to Z$ in $\mathbf{C}$, for a coalgebra $c\colon X \to GTX$, is obtained via the following three steps:*

1. *Transform c into a $\widehat{G}$-coalgebra $\mathcal{F}_{\mathcal{EM}}(c)$.*
2. *Get the resulting $\widehat{G}$-coalgebra map $TX \to Z$ in $\mathcal{EM}(T)$ by finality.*
3. *Precompose this map with the unit, yielding $X \to TX \to Z$ in $\mathbf{C}$.*

Note that Lemma 1 enables Step 1. Also, recall that the final G-coalgebra $Z \xrightarrow{\cong} GZ$ lifts to a final $\widehat{G}$-coalgebra by Proposition 3, which enables Step 2. Intuitively, Step 1 in Definition 1 corresponds to constructing the determinization of the original coalgebra; Step 2 provides semantics (for the determinization) via finality; Step 3 extracts the semantics for individual states. The reader will find elaborated examples of extension semantics in Section 5.

The next two sections will introduce examples of $\mathcal{EM}$-laws. Here, we briefly look at $\mathcal{K}\ell$-laws. The following result, from [30], shows that these $\mathcal{K}\ell$-laws are quite common, namely for *commutative* monads and *analytic functors*.

Lemma 2. *Let $T\colon \mathbf{Sets} \to \mathbf{Sets}$ be a commutative monad, and $F\colon \mathbf{Sets} \to \mathbf{Sets}$ an analytic functor. Then there is a (canonical) $\mathcal{K}\ell$-law $\lambda\colon FT \Rightarrow TF$.* □

4. Deterministic automata

This section briefly describes deterministic automata as coalgebras, recalls the final coalgebra, and introduces an associated $\mathcal{EM}$-law.

For arbitrary sets A, B there is an endofunctor $B \times (-)^A\colon \mathbf{Sets} \to \mathbf{Sets}$. Its coalgebras $\phi = \langle \phi_o, \phi_i \rangle\colon X \to B \times X^A$ are deterministic (Moore) automata. The map $\phi_o\colon X \to B$ describes the immediate output. The map $\phi_i\colon X \to X^A$ is the transition function, mapping a state $x \in X$ and an input $a \in A$ to a successor state $\phi_i(x)(a) \in X$. For the special case $B = 2 = \{0, 1\}$, the map ϕ_o tells of a state whether it is final or not. The following result is standard, so we omit the proof.

Lemma 3. *The final coalgebra of the functor $B \times (-)^A$ on $\mathbf{Sets}$ is given by the set of functions $B^{A^\star}$, with structure:*

$$B^{A^\star} \xrightarrow{\zeta = \langle \zeta_o, \zeta_i \rangle} B \times \big(B^{A^\star}\big)^A$$

defined via the empty sequence $\langle\rangle \in A^\star$ and via prefixing $a \cdot \sigma$ of $a \in A$ to $\sigma \in A^\star$: For $t \in B^{A^\star}$

$$\zeta_o(t) \;=\; t(\langle\rangle) \qquad \text{and} \qquad \zeta_i(t)(a) \;=\; \lambda\sigma \in A^\star.\, t(a \cdot \sigma).$$ □

This result captures the paradigm of trace semantics: a state $x \in X$ of an arbitrary coalgebra $X \to B \times X^A$ has a "behaviour" in the carrier of the final coalgebra $B^{A^\star}$ that maps a trace-as-word of inputs in $A^\star$ to an output in B. In the sequel we consider the special case where the output set is the free algebra $T(B)$, for a monad T. In the next result we show that we then get a distributive law. We apply this result only for the category $\mathbf{Sets}$. But it will be formulated more generally, using a strong monad [27] on a Cartesian closed category. This strength is automatic for any monad on $\mathbf{Sets}$ [27].

The following property follows directly from the properties of an alternative formulation of strength $\mathrm{st}\colon T(U^V) \to T(U)^V$, given by $\mathrm{st}(h)(x) = T\big(\lambda p.\, p(x)\big)(h)$ for $h \in T(U^V)$ and $x \in V$. Such a formulation of strength [22, Exercise 4.8.13] arises from the monad strength st_T by $\mathrm{st} = T(\mathrm{ev})^A \circ \Lambda(\mathrm{st}_T)$ where $\mathrm{ev}\colon A \times B^A \to B$ is the evaluation map and $\Lambda(f)\colon A \to B^C$, for an arrow $f\colon A \times C \to B$, is the abstraction map.

Lemma 4. *Let T be a strong monad on a Cartesian closed category $\mathbf{C}$ and let $A, B \in \mathbf{C}$ be arbitrary objects. Consider the associated "machine" endofunctor $M = T(B) \times (-)^A$ on $\mathbf{C}$. Then there is an $\mathcal{EM}$-law $\rho\colon TM \Rightarrow MT$ given by:*

$$\rho_X = \Big(T\big(T(B) \times X^A\big) \xrightarrow{\langle T(\pi_1), T(\pi_2)\rangle} T^2(B) \times T(X^A) \xrightarrow{\mu \times \mathrm{st}} T(B) \times T(X)^A\Big).$$

□

This $\mathcal{EM}$-law can be defined slightly more generally, not with a free algebra $T(B)$ as output, but with an arbitrary algebra. But in fact, most of our examples involve free algebras.

The resulting lifting $\widehat{M}\colon \mathcal{EM}(T) \to \mathcal{EM}(T)$ sends an algebra $a\colon TX \to X$ to the algebra $TMX \to MX$ given by:

$$T\big(T(B) \times X^A\big) \xrightarrow{\langle T(\pi_1), T(\pi_2)\rangle} T^2(B) \times T(X^A) \xrightarrow{\mu \times (a^A \circ \mathrm{st})} T(B) \times X^A$$

Proposition 3, in combination with Lemma 3, says that the final M-coalgebra $T(B)^{A^\star}$ carries a T-algebra structure that forms the final $\widehat{M}$-coalgebra in $\mathcal{EM}(T)$. With a bit of effort one shows that this algebra on $T(B)^{A^\star}$ is given by:

$$\alpha = \Big(T\big(T(B)^{A^\star}\big) \xrightarrow{\text{st}} \big(T^2(B)\big)^{A^\star} \xrightarrow{\mu^{A^\star}} T(B)^{A^\star}\Big)$$

Then, by Proposition 3, the map $\zeta\colon \alpha \xrightarrow{\cong} \widehat{M}(\alpha)$ from Lemma 3 forms the final $\widehat{M}$-coalgebra.

4.1. Finality via duality

We have just seen how to obtain a final coalgebra in categories of Eilenberg-Moore algebras, for the lifted automaton functor $B \times (-)^A$. It turns out that in the particular cases that we are interested in there is an alternative way to obtain such finality, namely via duality. This will be described in the current subsection; it involves a very limited class of functors, namely of the form $F = B + A \times (-)$ where B, A are sets. The results below are not needed for what follows. This duality-based approach makes a connection to the work of [34, 26].

We start from a basic result, given as exercise in [22].

Lemma 5. *Let $T\colon \mathbf{Sets} \to \mathbf{Sets}$ be a monad, and $d\colon T(D) \to D$ an arbitrary (but fixed) Eilenberg-Moore algebra. Then there is an adjunction:*

$$\mathbf{Sets}^{op} \underset{Hom(-,d)}{\overset{D^{(-)}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\top}}}} \mathcal{EM}(T) \tag{7}$$

PROOF. Since algebras are closed under products, for each set X there is an algebra structure on D^X given by:

$$T\big(D^X\big) \xrightarrow{\text{st}} T(D)^X \xrightarrow{d^X} D^X,$$

where st is the strength map (as in Lemma 4). It is easy to see that for functions $f\colon X \to Y$ and $\varphi\colon Y \to D$ the map $D^f(\varphi) = \varphi \circ f\colon D^Y \to D^X$ is a homomorphism of algebras. Thus we have a functor $\mathbf{Sets}^{\text{op}} \to \mathcal{EM}(T)$.

In the other direction we map an algebra $\beta\colon T(B) \to B$ to the set

$$Hom(\beta, d) = \{g\colon B \to D \mid g \circ d = \beta \circ T(g)\}.$$

This obviously yields a functor $\mathcal{EM}(T) \to \mathbf{Sets}^{\text{op}}$. Moreover, there is a bijective correspondence:

$$\frac{\begin{pmatrix} T(B) \\ {\scriptstyle\beta}\downarrow \\ B \end{pmatrix} \xrightarrow{\ f\ } \begin{pmatrix} T(D^X) \\ \downarrow{\scriptstyle d^X \circ \text{st}} \\ D^X \end{pmatrix}}{X \xrightarrow{\ g\ } Hom(\beta, d)}$$

It is obtained by swapping arguments.

- Given an algebra map $f\colon B \to D^X$ define $\overline{f}\colon X \to D^B$ by $\overline{f}(x)(b) = f(b)(x)$. We have to show that $\overline{f}(x)\colon B \to D$ is an algebra map. Well, for $u \in T(B)$:

$$\begin{aligned}
\big(\overline{f}(x) \circ \beta\big)(u) \;=\; \overline{f}(x)(\beta(u)) \;=\; f(\beta(u))(x) \;&=\; \big(d^X \circ \text{st} \circ T(f)\big)(u)(x) \\
&=\; d\Big(\text{st}\big(T(f)(u)\big)(x)\Big) \\
&=\; d\Big(T\big((\lambda p.\, p(x)) \circ f\big)(u)\Big) \\
&=\; d\Big(T\big(\lambda b.\, f(b)(x)\big)(u)\Big) \\
&=\; d\Big(T\big(\lambda b.\, \overline{f}(x)(b)\big)(u)\Big) \\
&=\; \big(d \circ T(\overline{f}(x))\big)(u).
\end{aligned}$$

- Similarly, for a function $g\colon X \to Hom(\beta,\alpha)$, the map $\overline{g}\colon B \to D^X$ defined by $\overline{g}(b)(x) = g(x)(b)$ forms a map of algebras.

It is easy to see that these constructions are inverse to each other. □

We need another basic result about lifting adjunctions in situations with a "dual" adjunction $\mathbf{A}^{\mathrm{op}} \leftrightarrows \mathbf{B}$, like in [24] and many other places.

Lemma 6. *Consider the situation:*

$$H \circlearrowright \mathbf{C}^{op} \underset{F}{\overset{P}{\rightleftarrows}} \mathbf{A} \circlearrowleft K \qquad \text{with} \quad F \dashv P, \text{ unit } \eta, \text{ and counit } \varepsilon \tag{8}$$

We assume a natural transformation $\sigma\colon PH \Rightarrow KP$.

1. *This σ gives rise to a functor $\overline{P}\colon \mathbf{Alg}(H)^{op} \to \mathbf{CoAlg}(K)$, given by:*

$$\overline{P}\Big(H(B) \xrightarrow{\beta} B\Big) \;=\; \Big(P(B) \xrightarrow{P(\beta)} PH(B) \xrightarrow{\sigma_B} KP(B)\Big).$$

2. *If this $\sigma\colon PH \Rightarrow KP$ is an isomorphism, then there is a similar lifting of the functor $F\colon \mathbf{A} \to \mathbf{C}^{op}$ to $\overline{F}\colon \mathbf{CoAlg}(K) \to \mathbf{Alg}(H)^{op}$, by:*

$$\overline{F}\Big(X \xrightarrow{c} K(X)\Big) \;=\; \Big(HF(X) \xrightarrow[\cong]{\overline{\sigma}_X} FK(X) \xrightarrow{F(c)} F(X)\Big).$$

   where $\overline{\sigma}\colon HF \Rightarrow FK$ is obtained as:

$$\overline{\sigma} = \Big(HF \xRightarrow{\varepsilon HF} FPHF \xRightarrow{F\sigma^{-1}F} FKPF \xRightarrow{FK\eta} FK\Big).$$

3. *This yields an adjunction $\overline{F} \dashv \overline{P}$ over $F \dashv P$ in:*

$$\begin{array}{ccc} \mathbf{Alg}(H)^{op} & \underset{\overline{F}}{\overset{\overline{P}}{\rightleftarrows}} & \mathbf{CoAlg}(K) \\ \downarrow & & \downarrow \\ \mathbf{C}^{op} & \underset{F}{\overset{P}{\rightleftarrows}} & \mathbf{A} \end{array} \qquad H \circlearrowright \quad K \circlearrowleft$$

PROOF. By naturality of σ and $\overline{\sigma}$ these $\overline{F}$ and $\overline{P}$ are functors. We need to show that there is a bijective correspondence between homomorphisms of (co)algebras f and g in:

$$\begin{array}{ll} \dfrac{\overline{F}\Big(X \xrightarrow{c} K(X)\Big) \xrightarrow{f} \Big(H(B) \xrightarrow{\beta} B\Big)}{\Big(X \xrightarrow{c} K(X)\Big) \xrightarrow{g} \overline{P}\Big(H(B) \xrightarrow{\beta} B\Big)} & \begin{array}{l}\text{in } \mathbf{Alg}(H)^{\mathrm{op}} \\ \\ \text{in } \mathbf{CoAlg}(K)\end{array} \end{array}$$

This correspondence is essentially the one of the underlying adjunction $F \dashv P$. □

Lemma 7. *Let T be a monad with algebra $d\colon T(D) \to D$ giving rise to an adjunction* (7)*. For arbitrary sets A, B consider the two functors $F, G\colon \mathbf{Sets} \to \mathbf{Sets}$ given by:*

$$F(X) \;=\; B + (A \times X) \qquad \text{and} \qquad G(X) \;=\; D^B \times X^A.$$

1. *There is an $\mathcal{EM}$-law $\rho\colon TG \Rightarrow GT$, with components:*

$$\rho_X = \Big(T(D^B \times X^A) \xrightarrow{\langle T(\pi_1), T(\pi_2)\rangle} T(D^B) \times T(X^A) \\ \qquad\qquad \downarrow{\scriptstyle \mathrm{st}\times\mathrm{st}} \\ T(D)^B \times T(X)^A \xrightarrow{d^B \times id} D^B \times T(X)^A \Big).$$

2. *This ρ induces a lifting $\widehat{G}\colon \mathcal{EM}(T) \to \mathcal{EM}(T)$ for which there is an isomorphism $\sigma\colon D^{(-)}F \Rightarrow \widehat{G}D^{(-)}$.*
3. *Lemma 6 now makes it possible to lift the adjunction* (7) *in:*

$$\begin{array}{ccc} \mathbf{Alg}(F)^{op} & \underset{}{\overset{\top}{\rightleftarrows}} & \mathbf{CoAlg}(\widehat{G}) \\ \downarrow & & \downarrow \\ \mathbf{Sets}^{op} & \underset{Hom(-,d)}{\overset{D^{(-)}}{\rightleftarrows}} & \mathcal{EM}(T) \\ {\scriptstyle F}\circlearrowleft & & \circlearrowright{\scriptstyle \widehat{G}} \end{array} \tag{9}$$

PROOF. We leave it to the reader to verify that ρ as defined above is indeed a distributive law. The isomorphism $\sigma_X\colon D^{F(X)} \cong G(D^X)$ in the second point is simply:

$$D^{F(X)} \;=\; D^{B+(A\times X)} \;\cong\; D^B \times D^{A\times X} \;\cong\; D^B \times \big(D^X\big)^A \;=\; G(D^X).$$

Some elementary calculations show that it is a map of algebras. The lifting of adjunctions follows directly from Lemma 6. □

Many of the trace semantics examples are instances of the following result.

Theorem 1. *Let $T\colon \mathbf{Sets} \to \mathbf{Sets}$ be a monad with dual adjunction $\mathbf{Sets}^{op} \leftrightarrows \mathcal{EM}(T)$ as in* (7)*, resulting from an algebra $d\colon T(D) \to D$. For each functor $F = B + (A \times (-))$ we consider the associated functor $G = D^B \times (-)^A$ with its lifting $\widehat{G}\colon \mathcal{EM}(T) \to \mathcal{EM}(T)$ as in Lemma 6. The initial F-algebra $B \times A^\star$ in $\mathbf{Sets}$ then yields the final $\widehat{G}$-coalgebra $D^{B\times A^\star}$ in $\mathcal{EM}(T)$.*

PROOF. The initial object in $\mathbf{Alg}(F)$ is final in the dual category $\mathbf{Alg}(F)^{\mathrm{op}}$. Hence it is preserved by the right adjoint $\mathbf{Alg}(F)^{\mathrm{op}} \to \mathbf{CoAlg}(\widehat{G})$ in (9). This right adjoint sends the carrier $B \times A^\star$ of the initial F-algebra to the carrier $D^{B\times A^\star}$ of the final $\widehat{G}$-coalgebra. □

5. Examples of $\mathcal{EM}$-extension semantics

This section describes three examples of $\mathcal{EM}$-laws, one familiar one in the context of non-deterministic automata, a new one for simple Segala systems. The latter involves a more general law (Lemma 8) that can also be used for alternating automata and non-deterministic multiset automata. Finally, we also include pushdown automata.

5.1. Non-deterministic automata, in $\mathcal{EM}$-style

We briefly restate the non-deterministic automaton example from [36], but this time using the general constructions developed so far. A non-deterministic automaton is understood here as a coalgebra $c\colon X \to 2 \times (\mathcal{P}X)^A$, which is of the form $X \to G(TX)$, where G is the functor $2 \times (-)^A$ and T is the powerset monad $\mathcal{P}$ on $\mathbf{Sets}$.

Since $2 = \{0, 1\}$ is the (carrier of the) free algebra $\mathcal{P}(1)$ on the singleton set $1 = \{*\}$, Lemma 4 applies. It yields an $\mathcal{EM}$-law with components $\rho = \langle \rho_1, \rho_2\rangle\colon \mathcal{P}(2 \times X^A) \to 2 \times (\mathcal{P}X)^A$, given by:

$$\begin{cases} \rho_1(U) = 1 \iff \exists \langle b, h\rangle \in U.\, b = 1 \\ x \in \rho_2(U)(a) \iff \exists \langle b, h\rangle \in U.\, h(a) = x. \end{cases}$$

Lemma 1 yields a coalgebra $\mathcal{F}_{\mathcal{EM}}(c) = \langle \phi_o, \phi_i \rangle \colon \mathcal{P}(X) \to 2 \times (\mathcal{P}X)^A$ in the category $\mathcal{EM}(\mathcal{P})$, where:

$$\begin{cases} \phi_o(U) = 1 \iff \exists x \in U.\, \pi_1 c(x) = 1 \\ y \in \phi_i(U)(a) \iff \exists x \in U.\, y \in \pi_2 c(x)(a). \end{cases}$$

By Proposition 3 the final G-coalgebra $2^{A^\star} = \mathcal{P}(A^\star)$ of languages is also final for the functor $\widehat{G}$ on $\mathcal{EM}(\mathcal{P})$. Hence, we get a map $[\![-]\!] \colon \mathcal{P}(X) \to \mathcal{P}(A^\star)$ by finality. Applying this map to the singleton set $\{x\} \in \mathcal{P}(X)$ yields the set of words that are accepted in the state $x \in X$. Thus, the $\mathcal{EM}$-semantics from Definition 1 yields the trace semantics for a non-deterministic automaton $c \colon X \to 2 \times (\mathcal{P}X)^A$, via the language accepted in a state.

To illustrate that different $\mathcal{EM}$-laws generate different semantics, consider the following alternative law $\rho' = \langle \rho_1', \rho_2 \rangle$, which is a slight adaptation of the law ρ above, namely by changing the existential quantification in ρ_1 to a universal one.

$$\rho_1{}'(U) = 1 \iff \forall \langle b, h \rangle \in U.\, b = 1$$

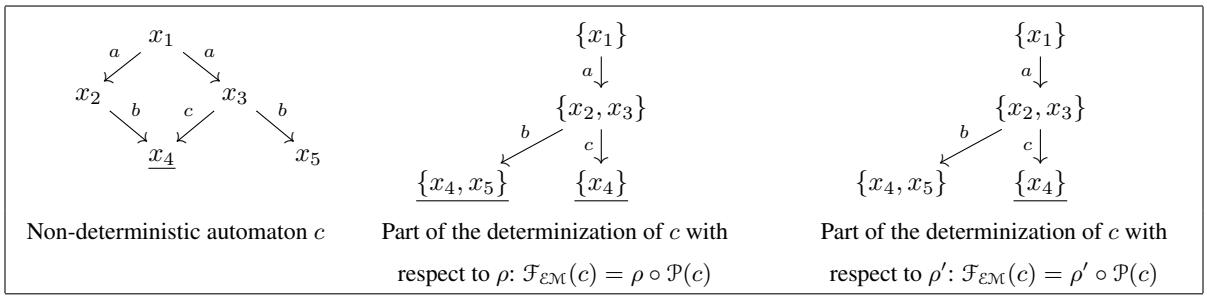
The determinization of $c \colon X \to 2 \times (\mathcal{P}X)^A$ is now given by $\mathcal{F}_{\mathcal{EM}}(c) = \langle \phi_o', \phi_i \rangle$, where

$$\phi_o'(U) = 1 \iff \forall x \in U.\, \pi_1 c(x) = 1.$$

and ϕ_i is defined as above. The map $[\![-]\!] \colon \mathcal{P}(X) \to \mathcal{P}(A^\star)$ into the final coalgebra yields, when applied to the singleton set $\{x\} \in \mathcal{P}(X)$, the set of words which, starting from x in the automaton $c \colon X \to 2 \times (\mathcal{P}X)^A$, always lead to a final state. That is,

$$w \in [\![\{x\}]\!] \iff \text{all paths starting from } x \text{ and labelled by } w \text{ are accepting in the automaton } c$$

Consider the following non-deterministic automaton $c \colon X \to 2 \times (\mathcal{P}X)^A$, with $X = \{x_1, x_2, x_3, x_4, x_5\}$ and only one final state x_4, which we underline. Starting from state $\{x_1\}$, the reachable parts of the two determinizations, arising from ρ and ρ' are given as follows.



Non-deterministic automaton c | Part of the determinization of c with respect to ρ: $\mathcal{F}_{\mathcal{EM}}(c) = \rho \circ \mathcal{P}(c)$ | Part of the determinization of c with respect to ρ': $\mathcal{F}_{\mathcal{EM}}(c) = \rho' \circ \mathcal{P}(c)$

Note the subtle difference: in the automaton displayed in the middle the state $\{x_1\}$ accepts the language $\{ab, ac\}$, whereas in the automaton depicted on the right the word ab is not accepted, since it also labels a non-accepting path in the automaton c: $x_1 \xrightarrow{a} x_3 \xrightarrow{b} x_5$.

5.2. Simple Segala systems, in $\mathcal{EM}$-style

Next, we consider simple Segala systems as a non-trivial example of $\mathcal{EM}$-extension semantics, which was not considered in [36]. These systems are also called simple probabilistic automata [35], Markov decision processes, or probabilistic labelled transition systems (LTSs). They are coalgebras of the form $c \colon X \to \mathcal{P}(A \times \mathcal{D}X)$, mixing probability and non-determinism. In a recent paper [13], with ideas appearing already in [33, 15, 10, 19], it has been recognized that it might be useful for verification purposes to transform them into so-called distribution LTSs, i.e. into LTSs with state space $\mathcal{D}X$ of so-called uncertain or belief states.

Given a simple Segala system $c \colon X \to \mathcal{P}(A \times \mathcal{D}X)$, we denote by $c^\sharp \colon \mathcal{D}X \to \mathcal{P}(A \times \mathcal{D}X)$ its distribution LTS from [13]. It is defined by

$$\begin{aligned} c^\sharp(\varphi) &= \{\langle a, \psi \rangle \mid \exists x \in \operatorname{supp}(\varphi).\, \langle a, \psi \rangle \in c(x)\} \\ &= \big(\mu^{\mathcal{P}} \circ \mathcal{P}(c) \circ \operatorname{supp}\big)(\varphi). \end{aligned} \tag{10}$$

where $\mathrm{supp}(\varphi) = \{x \in X \mid \varphi(x) \neq 0\}$. In the usual notation for transitions, this means that for $\varphi, \psi \in \mathcal{D}X$,

$$\varphi \xrightarrow{a}_{c^\sharp} \psi \qquad \text{if and only if} \qquad \exists x \in \mathrm{supp}(\varphi).\, x \xrightarrow{a}_{c} \psi.$$

Here, we capture this situation via a distributive law $\rho\colon \mathcal{D}\mathcal{P}(A \times (-)) \Longrightarrow \mathcal{P}(A \times \mathcal{D})$. It is an $\mathcal{EM}$-law $\rho\colon TG \Rightarrow GT$ for $T = \mathcal{D}$ and $G = \mathcal{P}(A \times -)$ with the property that the $\mathcal{EM}$-extension from Lemma 1 turns a simple Segala system into its distribution LTS, see Proposition 4 below. Explicitly, for a distribution $\Phi \in \mathcal{D}\mathcal{P}(A \times X)$, we define

$$\begin{aligned} \rho(\Phi) &= \{\langle a, \delta_x\rangle \mid \exists U \in \mathrm{supp}(\Phi).\, \langle a, x\rangle \in U\} \\ &= \textstyle\bigcup_{U \in \mathrm{supp}(\Phi)} \{\langle a, \delta_x\rangle \mid \langle a, x\rangle \in U\} \\ &= \big(\mu^{\mathcal{P}} \circ \mathcal{P}^2(\mathrm{id} \times \eta^{\mathcal{D}}) \circ \mathrm{supp}\big)(\Phi). \end{aligned} \tag{11}$$

where δ_x is the Dirac distribution assigning probability 1 to x, i.e. $\delta_x = \eta(x)$. The fact that ρ is an $\mathcal{EM}$-law is an instance of the following result—using that the support forms a map of monads $\mathrm{supp}\colon \mathcal{D} \Rightarrow \mathcal{P}$.

Lemma 8. *Each map of monads $\sigma\colon T \Rightarrow S$ induces an $\mathcal{EM}$-law*

$$TS(A \times -) \Longrightarrow S(A \times T(-))$$

of the monad T *over the* functor $S(A \times -)$. *The components of ρ are given by:*

$$\rho_X = \Big(TS(A \times X) \xrightarrow{\sigma_{S(A\times -)X}} S^2(A \times X) \xrightarrow{S^2(id \times \eta^T_X)} S^2(A \times TX) \xrightarrow{\mu^S_{(A\times TX)}} S(A \times TX)\Big). \qquad \square$$

Remark 1. *As emphasized, ρ in* (11) *is a distributive law between the monad $\mathcal{D}$ and the* functor $\mathcal{P}(A \times -)$. *In particular for $A = 1$ it is a distributive law between the monad $\mathcal{D}$ and the* functor $\mathcal{P}$. *An important but subtle point is that ρ is not a distributive law between the monad $\mathcal{D}$ and the* monad $\mathcal{P}$. *There is no such distributive law as shown in [39]. The unit law for the powerset monad, required for a monad distributive law, does not hold for ρ with $A = 1$: $\rho \circ \mathcal{D}(\eta^{\mathcal{P}}) \neq \eta^{\mathcal{P}}\mathcal{D}$. Nevertheless, one can distribute probability over non-determinism, via ρ. The construction provides a non-standard LTS semantics to simple Segala systems, by first lifting these systems to distributions.*

The distribution LTS in (10) from [13] is an instance of our general framework.

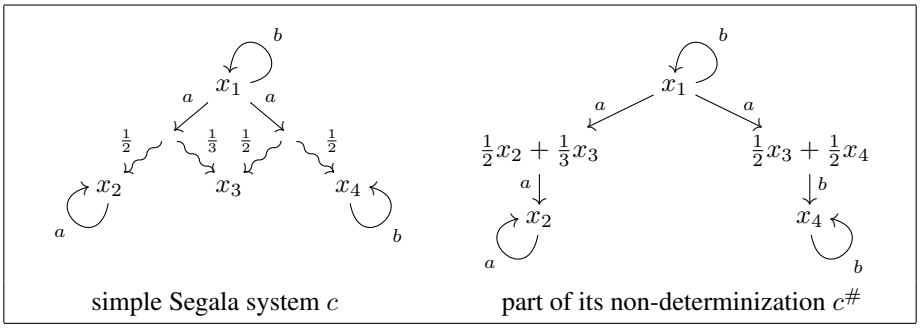
Proposition 4. *Given a simple Segala system $c\colon X \to \mathcal{P}(A \times \mathcal{D}X)$, its $\mathcal{EM}$-extension $\mathcal{F}_{\mathcal{EM}}(c)$ from Lemma 1, obtained via the $\mathcal{EM}$-law ρ from* (11)*, is the same as the coalgebra $c^\# : \mathcal{D}X \to \mathcal{P}(A \times \mathcal{D}X)$ described in Equation* (10)*.*

PROOF. By a straightforward calculation:

$$\begin{aligned} \mathcal{F}_{\mathcal{EM}}(c) &\overset{(6)}{=} \mathcal{P}(\mathrm{id} \times \mu^{\mathcal{D}}) \circ \rho \circ \mathcal{D}(c) \\ &\overset{(11)}{=} \mathcal{P}(\mathrm{id} \times \mu^{\mathcal{D}}) \circ \mu^{\mathcal{P}} \circ \mathcal{P}^2(\mathrm{id} \times \eta^{\mathcal{D}}) \circ \mathrm{supp} \circ \mathcal{D}(c) \\ &= \mathcal{P}(\mathrm{id} \times \mu^{\mathcal{D}}) \circ \mathcal{P}(\mathrm{id} \times \eta^{\mathcal{D}}) \circ \mu^{\mathcal{P}} \circ \mathrm{supp} \circ \mathcal{D}(c) \\ &= \mu^{\mathcal{P}} \circ \mathcal{P}(c) \circ \mathrm{supp} \\ &\overset{(10)}{=} c^\sharp. \end{aligned}$$

$\square$

The following is a simple example of a non-determinization.



simple Segala system c part of its non-determinization $c^\#$

In the representation of the non-determinization, on the right, state x_1 is formally $1x_1 = \eta(x_1)$, an element of $\mathcal{D}(X)$. Moreover, on the left, an arrow $x \xrightarrow{a} y$ means $x \xrightarrow{a} \overset{1}{\rightsquigarrow} y$.

Remark 2. *Definition 1 describes how $\mathcal{EM}$-extension semantics arises in presence of a final coalgebra. This does not directly apply in this situation because the (ordinary) powerset functor does not have a final coalgebra, for cardinality reasons. But if we restrict ourselves to finite subsets and distributions with finite support, there is still a map of monads $\mathcal{D}_{fin} \Rightarrow \mathcal{P}_{fin}$, so that we get an $\mathcal{EM}$-law $\mathcal{D}_{fin}\mathcal{P}_{fin}(A \times -) \Rightarrow \mathcal{P}_{fin}(A \times \mathcal{D}_{fin})$ by Lemma 8. For a "finitely branching" Segala system $c\colon X \to \mathcal{P}_{fin}(A \times \mathcal{D}_{fin}X)$ one obtains semantics $X \to Z$, where $Z \xrightarrow{\cong} \mathcal{P}_{fin}(A \times Z)$ is the final coalgebra.*

5.3. Alternating automata, in $\mathcal{EM}$-style

Alternating automata with only existential states and transitions [9, 11] are coalgebras of the form $c\colon X \to \mathcal{P}(A \times \mathcal{P}X)$ hence of type GT for $G = \mathcal{P}(A \times (-))$ and $T = \mathcal{P}$. As there is a trivial map of monads $\mathrm{id}\colon \mathcal{P} \Rightarrow \mathcal{P}$, Lemma 8 provides us with a distributive law

$$\rho\colon \mathcal{P}\mathcal{P}(A \times (-)) \Rightarrow \mathcal{P}(A \times \mathcal{P}(-))$$

with components, given concretely, by

$$\rho_X = \Big(\mathcal{P}\mathcal{P}(A \times X) \xrightarrow{\mathcal{P}\mathcal{P}(A \times \eta_X)} \mathcal{P}\mathcal{P}(A \times \mathcal{P}X) \xrightarrow{\mu_{(A \times \mathcal{P})X}} \mathcal{P}(A \times \mathcal{P}X)\Big) \tag{12}$$

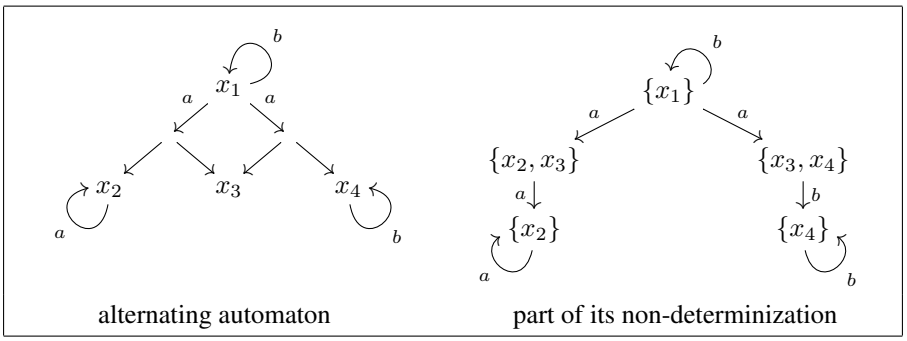
Given an alternating automaton $c\colon X \to \mathcal{P}(A \times \mathcal{P}X)$, the resulting $\mathcal{EM}$-extension $\mathcal{F}_{\mathcal{EM}}(c)\colon \mathcal{P}X \to \mathcal{P}(A \times \mathcal{P}X)$ from Lemma 1, obtained via the $\mathcal{EM}$-law ρ from (12), amounts to

$$\mathcal{F}_{\mathcal{EM}}(c) = G\mu \circ \rho \circ \mathcal{P}(c) = \mu \circ \mathcal{P}(c)$$

which in concrete terms gives

$$\mathcal{F}_{\mathcal{EM}}(c)(U) = \bigcup_{x \in U} c(x), \qquad \text{for } U \subseteq X.$$

Clearly, the $\mathcal{EM}$-extension (the non-determinization) of an alternating automaton is once again an LTS, this time over subsets of the original state space. This non-determinization looks pretty much the same as the one for simple Segala systems if one disregards the probabilities. Here is an example. In an alternating automaton, we use unlabelled arrows to denote non-deterministic branching (from the powerset monad) after an a-labelled transition.



alternating automaton part of its non-determinization

Remark 2 applies here as well, $\mathrm{id} : \mathcal{P}_{\text{fin}} \Rightarrow \mathcal{P}_{\text{fin}}$ is of course also a map of monads, leading to semantics $X \to Z$, where $Z \xrightarrow{\cong} \mathcal{P}_{\text{fin}}(A \times Z)$ is the final coalgebra, for a "finitely branching" alternating automaton $c\colon X \to \mathcal{P}_{\text{fin}}(A \times \mathcal{P}_{\text{fin}}X)$.

5.4. Non-deterministic weighted automata, in $\mathcal{EM}$-style

Coalgebras of the form $c\colon X \to \mathcal{P}(A \times \mathcal{M}_S X)$ are known as non-deterministic weighted automata or non-deterministic multiset automata [14, 17]. They allow non-deterministic choice over labelled transitions into a multiset with weights from a semiring S, commonly the semiring of natural numbers. These coalgebras are again of type GT for the functor $G = \mathcal{P}(A \times (-))$ and the monad $T = \mathcal{M}_S$.

Similar to the case of simple Segala systems, the support forms a map of monads $\mathrm{supp}\colon \mathcal{M}_S \Rightarrow \mathcal{P}$ and Lemma 8 provides us with a distributive law

$$\mathcal{M}_S\mathcal{P}(A \times (-)) \xRightarrow{\rho} \mathcal{P}(A \times \mathcal{M}_S(-))$$

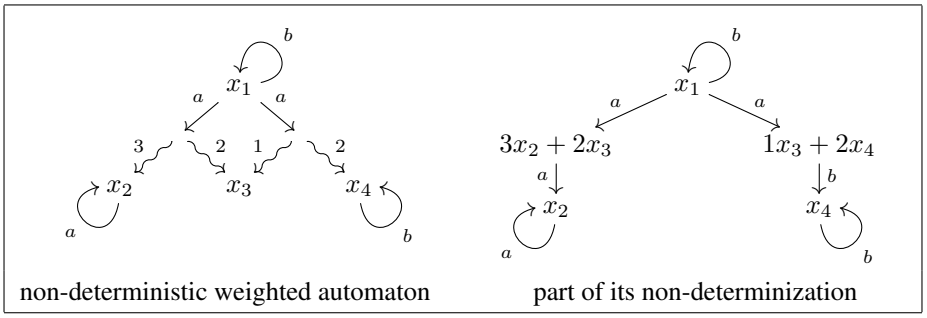
with components:

$$\rho_X \;=\; \mu^{\mathcal{P}} \circ \mathcal{P}\mathcal{P}(\mathrm{id} \times \eta^{\mathcal{M}_S}) \circ \mathrm{supp}_X\,.$$

Just like for simple Segala systems, this amounts to an $\mathcal{EM}$-extension $\mathcal{F}_{\mathcal{EM}}(c)\colon \mathcal{M}_S X \to \mathcal{P}(A \times \mathcal{M}_S X)$ (a non-determinization) of a non-deterministic weighted automaton $c\colon X \to \mathcal{P}(A \times \mathcal{M}_S X)$, given by

$$\mathcal{F}_{\mathcal{EM}}(c) \;=\; \mu^{\mathcal{P}} \circ \mathcal{P}(c) \circ \mathrm{supp}_X\,.$$

The only difference with the non-determinization of simple Segala systems is that probabilities are replaced by weights in a semiring. For completeness, we give an example with $S = \mathbb{N}$.



non-deterministic weighted automaton part of its non-determinization

Again, similar to the example of Segala systems, we denote, on the right, by x the formal sum $1x = \eta(x) \in \mathcal{M}_S(X)$. Also here Remark 2 applies as $\mathrm{supp}\colon \mathcal{M}_S \Rightarrow \mathcal{P}_{\text{fin}}$ is a map of monads, leading to LTS semantics for "finitely branching" non-deterministic weighted automata. Note that the multiset monad always has finite branching (finite support), so the finite branching requirement applies to the non-deterministic branching only: a finitely branching non-deterministic weighted automaton is one of the form $c\colon X \to \mathcal{P}_{\text{fin}}(A \times \mathcal{M}_S X)$.

5.5. Pushdown automata, in $\mathcal{EM}$-style

In this section we give another example of the framework by modeling pushdown automata coalgebraically, following [37], using the general constructions developed so far.

For this purpose, we will make use of $T(X) = \mathcal{P}(S \times X)^S$, the non-deterministic side-effect monad [5] with side-effects in a set S. The unit $\eta_X\colon X \to T(X)$ is given by $\eta(x)(s) = \{\langle s, x\rangle\}$, and the multiplication $\mu_X\colon TT(X)) \to T(X)$ is given, for $f \in \mathcal{P}(S \times (\mathcal{P}(S \times X)^S))^S$, by

$$\mu_X(f)(s) \;=\; \bigcup_{\langle s', g\rangle \in f(s)} g(s')$$

A (realtime) pushdown automaton (PDA) [1] is a tuple $(Q, \Sigma, \Gamma, \delta, \langle q_0, \alpha_0\rangle, C_F)$, where Q is the set of states, Σ is the set of input symbols, Γ is the set of stack symbols, $\delta\colon Q \to \mathcal{P}(Q \times \Gamma^\star)^{\Sigma\times\Gamma}$ is the transition function, $\langle q_0, \alpha_0\rangle \in Q \times \Gamma^\star$ is the initial configuration and $C_F \subseteq Q \times \Gamma^\star$ is the set of final configurations. Here, $Q \times \Gamma^\star$ is the set of all configurations.

This definition is slightly non-standard in the sense that no transition with empty word as input or empty stack is allowed, which is what "realtime" refers to. However, the definition is equivalent to the standard one, i.e., realtime PDA with the usual sets of accepting configurations accept the class of context-free languages, and it allows for a smooth coalgebraic treatment.

For convenience, we introduce the notion of transition relation for configurations. A configuration $\langle q, b\beta\rangle$ evolves to a configuration $\langle q', \alpha\beta\rangle$ by reading input $a \in \Sigma$, written $\langle q, b\beta\rangle \xrightarrow{a} \langle q', \alpha\beta\rangle$, if and only if $\langle q', \alpha\rangle \in \delta(q)(a, b)$. The transition relation extends to words $w \in \Sigma^\star$ in the usual way.

A word $w \in \Sigma^\star$ is accepted by a PDA if $\langle q_0, \alpha_0\rangle \xrightarrow{w} \langle q, \alpha\rangle$ and $\langle q, \alpha\rangle \in C_F$.

This definition captures several (equivalent) versions of acceptance for PDA. For instance, by taking $C_F = F \times \{\langle\rangle\}$, for $F \subseteq Q$, we obtain the so-called acceptance by final states and empty stack.

Every pushdown automaton induces a function $\langle o, t\rangle \colon Q \to GTQ$ where G is the functor $2^{\Gamma^\star} \times (-)^\Sigma$ and T is the non-deterministic side-effect monad defined above specialized for $S = \Gamma^\star$ (intuitively, side effects in a pushdown machine are changes in the stack). The functions $o \colon Q \to 2^{\Gamma^\star}$ and $t \colon Q \to (\mathcal{P}(Q \times \Gamma^*)^{\Gamma^\star})^\Sigma$ are defined as

$$\begin{aligned} o(q)(\beta) &= 1 \text{ if and only if } \langle q, \beta\rangle \in C_F \\ t(q)(a)(\langle\rangle) &= \emptyset \\ t(q)(a)(b\beta) &= \{\langle q', \alpha\beta\rangle \mid \langle q', \alpha\rangle \in \delta(q)(a, b)\} \end{aligned}$$

Hence, the output function o keeps track of final configuration and the transition function t describes the steps between PDA configurations as specified by the transition function δ. In particular $t(q)(a)(b) = \delta(q)(a, b)$.

Note that every pushdown automaton gives rise to a GT-coalgebra, but not every function $\langle o, t\rangle \colon Q \to GTQ$ defines a (realtime) pushdown automaton, since, for instance, $t(q)$ may depend on the content of the whole stack and not just on the top element.

We write $q \xrightarrow{a,b|\alpha} q'$ for $\langle q', \alpha\rangle \in t(q)(a)(b)$ indicating that the automaton is in the state q, reads an input symbol a, pops b from the stack, moves to state q' and pushes the string $\alpha \in \Gamma^\star$ on top of the stack.

There is an $\mathcal{EM}$-law $\rho \colon TG \Rightarrow GT$ with components:

$$\mathcal{P}(\Gamma^\star \times (2^{\Gamma^\star} \times Q^\Sigma))^{\Gamma^\star} \xrightarrow{\rho_Q = \langle \rho_1, \rho_2\rangle} 2^{\Gamma^\star} \times (\mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star})^\Sigma$$

given by:

$$\begin{cases} \rho_1(f)(\alpha) = 1 \iff \exists \langle \beta, \langle g, t\rangle\rangle \in f(\alpha).\, g(\beta) = 1 \\ \langle \beta, q\rangle \in \rho_2(f)(a)(\alpha) \iff \exists \langle \gamma, \langle g, t\rangle\rangle \in f(\alpha).\, \gamma = \beta \text{ and } q = t(a). \end{cases}$$

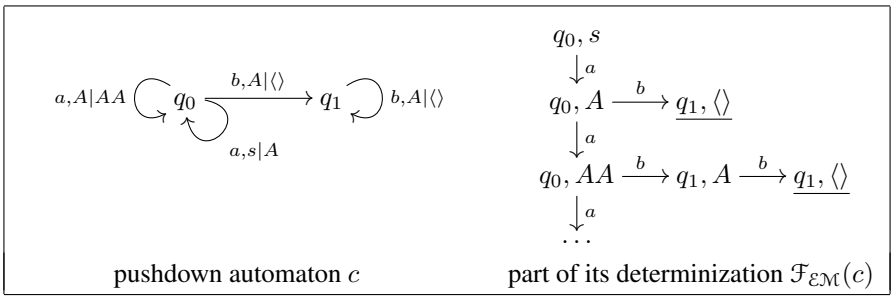
This law is an instance of the general law defined in Lemma 7, for $B = 1$ and $D = 2^{\Gamma^\star}$. Given a coalgebra $c \colon Q \to GTQ$, its $\mathcal{EM}$-extension $\mathcal{F}_{\mathcal{EM}}(c) \colon TQ \to GTQ$ is

$$\mathcal{F}_{\mathcal{EM}}(c) = (id \times \mu^\Sigma) \circ \rho \circ \mathcal{P}(id \times c)^{\Gamma^\star}.$$

More concretely, given such a coalgebra $c = \langle o, t\rangle$, we get $\mathcal{F}_{\mathcal{EM}}(c) = \langle o^\sharp, t^\sharp\rangle \colon \mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star} \to 2^{\Gamma^\star} \times (\mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star})^\Sigma$ as follows. For $f \in \mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star}$,

$$\begin{cases} o^\sharp(f)(\alpha) = 1 \iff \exists \langle \beta, q\rangle \in f(\alpha).\, o(q)(\beta) = 1 \\ \langle \beta, q\rangle \in t^\sharp(f)(a)(\alpha) \iff \exists \langle \gamma, r\rangle \in f(\alpha).\, \langle \beta, q\rangle \in t(r)(a)(\gamma). \end{cases}$$

The extension (determinization) can be thought of, in this case, as the infinite deterministic automaton that recognizes the same language. As an example consider the PDA below, on the left, with starting configuration $\langle q_0, s\rangle$ and set of final configurations $\{\langle q_1, \langle\rangle\rangle\}$. The language accepted by this PDA is $\{a^i b^i \mid i \geq 1\}$.



In the representation of the determinization above, on the right, we represent the state (function) $f \in TQ = \mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star}$ by its value when applied to the current stack, starting from $\eta(q_0)$. Hence, the initial state is represented by $\eta(q_0)(s) = \{\langle s, q_0\rangle\}$. We refrain from writing unnecessary brackets and swap the order of the PDA state and the stack content for better readability. This representation allows us to put in evidence that the determinization indeed is a representation of the infinite deterministic automaton that would recognize the language. Final states are, as before, underlined. Note that to enable this representation we actually use the bijective correspondence:

$$\begin{array}{c} \mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star} \longrightarrow 2^{\Gamma^\star} \times (\mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star})^{\Sigma} \\ \hline\hline \mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star} \longrightarrow 2^{\Gamma^\star} \times (\mathcal{P}(\Gamma^\star \times Q)^{\Sigma})^{\Gamma^\star} \\ \hline\hline \mathcal{P}(\Gamma^\star \times Q)^{\Gamma^\star} \times \Gamma^\star \longrightarrow 2 \times \mathcal{P}(\Gamma^\star \times Q)^{\Sigma} \end{array}$$

Clearly, $G = 2^{\Gamma^\star} \times (-)^{\Sigma}$ is a particular deterministic automata functor and its final coalgebra (see Lemma 3) is carried by $\left(2^{\Gamma^*}\right)^{\Sigma^*}$. It lifts to a final $\widehat{G}$-coalgebra in $\mathcal{EM}(T)$ by Proposition 3 and we get extension semantics by

$$Q \xrightarrow{\eta} TQ \xrightarrow{[\![-]\!]} \left(2^{\Gamma^*}\right)^{\Sigma^*}$$

where $[\![-]\!]$ is the unique map into the final.

The extension semantics allows us to recover the definition of acceptance for PDA. A word $w \in A^\star$ is accepted by a PDA-coalgebra with initial configuration $\langle q_0, \alpha_0\rangle$ if and only if $[\![\eta(q_0)]\!](w)(\alpha_0) = 1$. If one takes $C_F = F \times \{\langle\rangle\}$ and $\alpha_0 = \langle\rangle$, this definition coincides with the usual acceptance definition (via final states and empty stack): a word $w \in \Sigma^\star$ is accepted by a PDA-coalgebra if, starting from the initial state and empty stack, by reading w the PDA can reach a final state and an empty stack.

6. $\mathcal{K}\ell$-Extension semantics

In a next step we wish to develop extension semantics not only for coalgebras of the form $X \to G(TX)$, on the left in (1), but also for coalgebras $X \to T(FX)$, on the right in (1). This turns out to be more complicated. First of all, the lifting $\mathcal{F}_{\mathcal{K}\ell} \colon \mathbf{CoAlg}(TF) \to \mathbf{CoAlg}(\widehat{F})$ in Lemma 1 is not interesting in the current setting because it does not involve a state space extension $X \mapsto T(X)$.

The next thought might be to translate coalgebras $X \to T(FX)$ into coalgebras $X \to G(TX)$, via a distributive law $TF \Rightarrow GT$. This results in functors

$$\mathbf{CoAlg}(TF) \longrightarrow \mathbf{CoAlg}(GT) \xrightarrow{\text{Lemma 1}} \mathbf{CoAlg}(\widehat{G})$$

where the first functor is obtained by post-composing with the distributive law. We will get back to this point later. However, coalgebras $X \to T(FX)$ are usually considered as coalgebras $X \to \widehat{F}X$ of the lifted functor $\widehat{F}$ on the Kleisli category $\mathcal{K}\ell(T)$. Therefore, our aim is to obtain a functor $\mathbf{CoAlg}(\widehat{F}) \to \mathbf{CoAlg}(\widehat{G})$. This will be described below.

Recall from Section 2 that there is a comparison functor $E \colon \mathcal{K}\ell(T) \to \mathcal{EM}(T)$. In this section we show how it can be lifted to coalgebras. We consider the following situation.

1. A monad $T\colon \mathbf{C} \to \mathbf{C}$ on a category $\mathbf{C}$.
2. An endofunctor $F\colon \mathbf{C} \to \mathbf{C}$ with a $\mathcal{K}\ell$-law $\lambda\colon FT \Rightarrow TF$, leading to a lifting $\widehat{F}\colon \mathcal{K}\ell(T) \to \mathcal{K}\ell(T)$ to T's Kleisli category, via Proposition 1.
3. Another endofunctor $G\colon \mathbf{C} \to \mathbf{C}$, but this time with an $\mathcal{EM}$-law $\rho\colon TG \Rightarrow GT$, yielding a lifting $\widehat{G}\colon \mathcal{EM}(T) \to \mathcal{EM}(T)$ to the category of T-algebras.
4. An "extension" natural transformation $\mathfrak{e}\colon TF \Rightarrow GT$ that connects the $\mathcal{K}\ell$- and $\mathcal{EM}$-laws via the following two commuting diagrams.

$$\begin{array}{ccccc} TFT & \xrightarrow{T(\lambda)} & T^2F & \xrightarrow{\mu F} & TF \\ {\scriptstyle \mathfrak{e}T}\downarrow & & & & \downarrow{\scriptstyle \mathfrak{e}} \\ GT^2 & & \xrightarrow{\quad G\mu \quad} & & GT \end{array} \qquad\qquad \begin{array}{ccccc} T^2F & & \xrightarrow{\quad \mu F \quad} & & TF \\ {\scriptstyle T\mathfrak{e}}\downarrow & & & & \downarrow{\scriptstyle \mathfrak{e}} \\ TGT & \xrightarrow{\rho T} & GT^2 & \xrightarrow{G\mu} & GT \end{array} \tag{13}$$

When such $\mathfrak{e}$ is an isomorphism, it forms a "commuting pair of endofunctors" from [3].

Theorem 2. *Assuming the above points 1–4, there is a lifting $\widehat{E}$ of the extension functor E in:*

$$\begin{array}{ccc} \mathbf{CoAlg}(\widehat{F}) & \xrightarrow{\widehat{E}} & \mathbf{CoAlg}(\widehat{G}) \\ \downarrow & & \downarrow \\ \mathcal{K}\ell(T) & \xrightarrow{\quad E \quad} & \mathcal{EM}(T) \\ {\scriptstyle \widehat{F}}\circlearrowright & & \circlearrowleft{\scriptstyle \widehat{G}} \end{array}$$

This functor $\widehat{E}$ is automatically faithful; and it is also full if the extension natural transformation $\mathfrak{e}\colon TF \Rightarrow GT$ consists of monomorphisms.

PROOF. We define the functor $\widehat{E}\colon \mathbf{CoAlg}(\widehat{F}) \to \mathbf{CoAlg}(\widehat{G})$ by:

$$\begin{array}{rcl} \big(X \xrightarrow{c} \widehat{F}X\big) & \longmapsto & \big(TX \xrightarrow{T(c)} T^2FX \xrightarrow{\mu} TFX \xrightarrow{\mathfrak{e}} GTX\big) \\ f & \longmapsto & E(f) = \mu \circ T(f). \end{array}$$

We need to show that $\widehat{E}(c)$ is a map of algebras $E(X) = \mu_X \to \widehat{G}(\mu_X) = \widehat{G}(EX)$ in:

$$\begin{pmatrix} T^2X \\ \downarrow{\scriptstyle \mu_X} \\ TX \end{pmatrix} \xrightarrow{\widehat{E}(c)} \begin{pmatrix} TGTX \\ \downarrow{\scriptstyle G(\mu_X)\circ\rho} \\ GTX \end{pmatrix} = \widehat{G} \begin{pmatrix} T^2X \\ \downarrow{\scriptstyle \mu_X} \\ TX \end{pmatrix}$$

But this is simply the above requirement 4.

Assume f is a map of $\widehat{F}$-coalgebras, from $c\colon X \to \widehat{F}X$ to $d\colon Y \to \widehat{F}Y$. That is, $f\colon X \to TY$, $c\colon X \to TFX$ and $d\colon Y \to TFY$ satisfy:

$$\mu \circ T(d) \circ f \;=\; \mu \circ T(\lambda \circ F(f)) \circ c. \tag{14}$$

Then $E(f) = \mu \circ T(f)$ is a map of coalgebras $\widehat{E}(c) \to \widehat{E}(d)$, again by requirement 4:

$$\begin{array}{rcl} \widehat{E}(d) \circ E(f) & = & \mathfrak{e}_Y \circ \mu \circ T(d) \circ \mu \circ T(f) \\ & = & \mathfrak{e}_Y \circ \mu \circ \mu \circ T(T(d) \circ f) \\ & = & \mathfrak{e}_Y \circ \mu \circ T(\mu \circ T(d) \circ f) \\ & \overset{(14)}{=} & \mathfrak{e}_Y \circ \mu \circ T(\mu \circ T(\lambda \circ F(f)) \circ c) \\ & = & \mathfrak{e}_Y \circ \mu \circ \mu \circ T^2(\lambda \circ F(f)) \circ T(c) \\ & = & \mathfrak{e}_Y \circ \mu \circ T(\lambda \circ F(f)) \circ \mu \circ T(c) \\ & \overset{(13)}{=} & G(\mu) \circ \mathfrak{e}_{TY} \circ TF(f) \circ \mu \circ T(c) \\ & = & G(\mu \circ T(f)) \circ \mathfrak{e}_X \circ \mu \circ T(c) \\ & = & \widehat{G}(Ef) \circ \widehat{E}(c). \end{array}$$

Clearly, the functor $\widehat{E}$ is faithful: if $f, g\colon X \to TY$ satisfy $\widehat{E}(f) = E(f) = E(g) = \widehat{E}(g)$, then $f = E(f) \circ \eta = E(g) \circ \eta = g$.

Now assume the components $\mathfrak{e}_X\colon TFX \to GTX$ are monomorphisms in $\mathbf{C}$. Towards fulness of $\widehat{E}$, let $h\colon \widehat{E}(c) \to \widehat{E}(d)$ be a morphism in $\mathbf{CoAlg}(\widehat{G})$. It is a map $h\colon TX \to TY$ that is both a map of algebras and of coalgebras, so:

$$\begin{array}{ccc} T^2X & \xrightarrow{T(h)} & T^2Y \\ {\scriptstyle\mu}\downarrow & & \downarrow{\scriptstyle\mu} \\ TX & \xrightarrow[h]{} & TY \end{array} \qquad\qquad \begin{array}{ccc} GTX & \xrightarrow{G(h)} & GTY \\ {\scriptstyle\mathfrak{e}_X\circ\mu\circ T(c)}\uparrow & & \uparrow{\scriptstyle\mathfrak{e}_Y\circ\mu\circ T(d)} \\ TX & \xrightarrow[h]{} & TY \end{array} \tag{15}$$

We now take $f = h \circ \eta\colon X \to T(Y)$ and need to show that $\widehat{E}(f) = E(f) = h$ and that it is a map of $\widehat{F}$-coalgebras $c \to d$. The first is easy since:

$$E(f) \;=\; \mu \circ T(h \circ \eta) \;=\; h \circ \mu \circ T(\eta) \;=\; h.$$

In order to show that f is a map of coalgebras we use that $\mathfrak{e}$ consists of monos, in:

$$\begin{array}{rcl} \mathfrak{e} \circ \big(d \odot f\big) & = & \mathfrak{e} \circ \mu \circ T(d) \circ f \\ & = & \mathfrak{e} \circ \mu \circ T(d) \circ h \circ \eta \\ & \overset{(15)}{=} & G(h) \circ \mathfrak{e} \circ \mu \circ T(c) \circ \eta \\ & = & G(h) \circ \mathfrak{e} \circ \mu \circ \eta \circ c \\ & = & G(h) \circ \mathfrak{e} \circ c \\ & = & G(h) \circ G(\mu \circ T(\eta)) \circ \mathfrak{e} \circ c \\ & \overset{(15)}{=} & G(\mu) \circ GT(h \circ \eta) \circ \mathfrak{e} \circ c \\ & = & G(\mu) \circ \mathfrak{e} \circ TF(f) \circ c \\ & \overset{(13)}{=} & \mathfrak{e} \circ \mu \circ T(\lambda \circ F(f)) \circ c \\ & = & \mathfrak{e} \circ \big(\widehat{F}(f) \odot c\big). \end{array}$$

□

On a more abstract level, what the previous result does is lift $\mathfrak{e}\colon TF \Rightarrow GT$ to a natural transformation $\widehat{\mathfrak{e}}\colon E\widehat{F} \Rightarrow \widehat{G}E$. In this way we can also define the functor $\widehat{E}\colon \mathbf{CoAlg}(\widehat{F}) \to \mathbf{CoAlg}(\widehat{G})$ by:

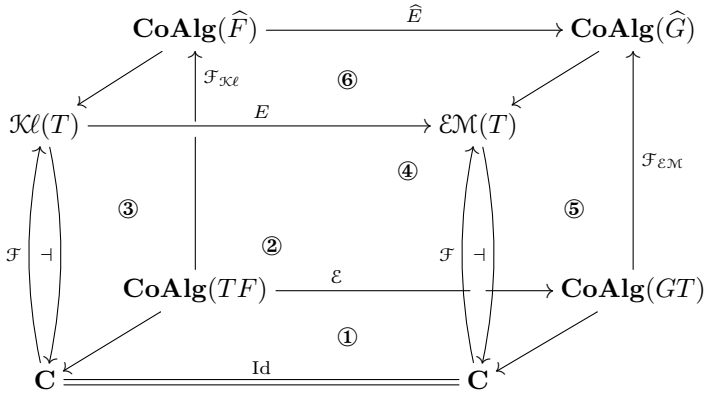
$$\big(X \xrightarrow{c} \widehat{F}X\big) \;\longmapsto\; \big(EX \xrightarrow{\widehat{\mathfrak{e}}\circ E(c)} \widehat{G}(EX)\big) \qquad \text{and} \qquad f \;\longmapsto\; E(f) = \mu \circ T(f).$$

Getting back to the original intention, let $\mathcal{E}\colon \mathbf{CoAlg}(TF) \to \mathbf{CoAlg}(GT)$ be the functor obtained by post-composing with the extension natural transformation $\mathfrak{e}$, that is, given $c\colon X \to TF(X)$ we have

$$\mathcal{E}(c) = \mathfrak{e}_X \circ c \quad \text{and} \quad \mathcal{E}(f) = f$$

for a coalgebra homomorphism f. We can now summarize all results in one cube-shaped diagram.

Theorem 3. *Under the assumptions 1. - 4. from the beginning of this section, we have the following commuting cube of (lifted) functors:*



PROOF. The bottom and frontal faces – ① and ② – commute trivially as $\mathcal{E}$ and E are functors. Lemma 1 provides the commutativity of the left and right faces – ③ and ⑤ – under the existence of a $\mathcal{K}\ell$-law and $\mathcal{EM}$-law, respectively. The back face ④ commutes by the definitions: we have, on objects,

$$\begin{aligned}(\mathcal{F}_{\mathcal{EM}} \circ \mathcal{E})(c) &= \mathcal{F}_{\mathcal{EM}}(\mathfrak{e} \circ c) \\ &= G\mu \circ \rho \circ T\mathfrak{e} \circ Tc \\ &\overset{(13)}{=} \mathfrak{e} \circ \mu \circ Tc \\ &= \widehat{E}(c) \\ &= (\widehat{E} \circ \mathcal{F}_{\mathcal{K}\ell})(c)\end{aligned}$$

and on morphisms,

$$\mathcal{F}_{\mathcal{EM}} \circ \mathcal{E}(f) = \mathcal{F}_{\mathcal{EM}}(f) = Tf = \mu \circ T\eta \circ Tf = \mu \circ T(\eta \circ f) = \widehat{E} \circ \mathcal{F}_{\mathcal{K}\ell}(f).$$

Commutativity of the top face ⑥ is given by Theorem 2. The extension law $\mathfrak{e}$ is needed for ④ and ⑥. □

We can now define extension semantics for coalgebras $X \to T(FX)$, analogously to Definition 1. The reader can find examples of $\mathcal{K}\ell$-extension semantics in Section 7.

Definition 2 ($\mathcal{K}\ell$-extension). *In addition to the points 1–4 from the beginning of this section, let the functor G in point 3. have a final coalgebra $Z \xrightarrow{\cong} G(Z)$. For a coalgebra $c\colon X \to T(FX)$ one obtains its $\mathcal{K}\ell$-extension semantics in three steps, like in Definition 1:*

1. *Transform c into a $\widehat{G}$-coalgebra $\widehat{E}(c)$.*
2. *Obtain a map $TX \to Z$ in $\mathcal{EM}(T)$ by finality.*
3. *Get $X \to Z$ by precomposition with the unit $X \to TX$.*

Step 1 is justified by Theorem 2. Note that Theorem 3 states that the same transformation is obtained via $\mathcal{F}_{\mathcal{EM}} \circ \mathcal{E}(c)$ since $\widehat{E}(c) = \widehat{E} \circ \mathcal{F}_{\mathcal{K}\ell}(c) = \mathcal{F}_{\mathcal{EM}} \circ \mathcal{E}(c)$. Step 2 is justified by Proposition 3, as the final G-coalgebra $Z \xrightarrow{\cong} G(Z)$ lifts to a final $\widehat{G}$-coalgebra. The intuition behind each of the steps is as follows: Step 1 produces a "determinization" after transforming the original TF-coalgebra to a GT-coalgebra; Step 2 provides semantics via finality, for the determinization; finally Step 3 gives the semantics of individual states.

We conclude this section by showing how the "Kleisli" trace semantics from [18] fits in the current setting. Thus, we assume in the situation of Definition 2 that the functor F has an initial algebra $\iota\colon F(W) \xrightarrow{\cong} W$.

Proposition 5. *Assume the map $\mathcal{F}(\iota^{-1})\colon W \to \widehat{F}(W)$ is final $\widehat{F}$-coalgebra. Each coalgebra $c\colon X \to TF(X)$ then gives rise to a "Kleisli" trace map in the Kleisli category (as in [18]), namely:*

$$\begin{array}{ccc} \widehat{F}X & \dashrightarrow & \widehat{F}(W) \\ {\scriptstyle c}\uparrow & & \cong\uparrow{\scriptstyle \mathcal{F}(\iota^{-1})} \\ X & \overset{\mathrm{tr}_{\mathcal{K}\ell}(c)}{\dashrightarrow} & W \end{array}$$

When we apply the functor $\widehat{E}$ from Theorem 2 to this diagram we get the rectangle on the left in:

$$\begin{array}{ccccc} & & \widehat{G}(TX) & \longrightarrow & \widehat{G}(TW) \dashrightarrow \widehat{G}(Z) \\ & & {\scriptstyle \widehat{E}(c)}\uparrow & & \cong \uparrow {\scriptstyle \widehat{E}(\mathcal{F}(\iota^{-1}))} \qquad \cong \uparrow {\scriptstyle \zeta} \\ X & \xrightarrow{\eta} & TX & \xrightarrow{\widehat{E}(\mathrm{tr}_{\mathcal{K}\ell}(c))} & TW \dashrightarrow Z \\ & & & \searrow^{\mathrm{tr}_{\mathcal{K}\ell}(c)} & \end{array} \tag{16}$$

The resulting $\mathcal{K}\ell$-extension semantics map $X \to Z$ is then the $\mathcal{K}\ell$-extension semantics of the final $\widehat{F}$-coalgebra $\mathcal{F}(\iota^{-1})\colon W \to \widehat{F}(W)$, precomposed with the Kleisli trace semantics $\mathrm{tr}_{\mathcal{K}\ell}(c)$. □

7. Determinization and trace semantics

In this section we specialize the $\mathcal{K}\ell$-extension framework developed so far to deterministic automata, leading to a novel definition of trace semantics, namely via $\mathcal{K}\ell$-extension semantics. Several illustrations will be given, including the standards ones from the trace semantics of [18].

Definition 3. *Let T be a monad and F an endofunctor on the category $\mathbf{Sets}$, with a $\mathcal{K}\ell$-law $\lambda\colon FT \Rightarrow TF$ between them. Let A, B be sets, leading to endofunctor $M = T(B) \times (-)^A$, which comes with an $\mathcal{EM}$-law $\rho\colon TM \Rightarrow MT$ like in Lemma 4. Finally, we assume an extension law $\mathfrak{e}\colon TF \Rightarrow MT = T(B) \times T(-)^A$ like in the previous section. In this situation,*

- *the determinization of a coalgebra $c\colon X \to TFX$ is the $\widehat{M}$-coalgebra $\widehat{E}(c)$ in the category of algebras $\mathcal{EM}(T)$ given by:*

$$T(X) \xrightarrow{T(c)} T^2FX \xrightarrow{\mu} TFX \xrightarrow{\mathfrak{e}} T(B) \times T(X)^A$$

  Thus, determinization turns the coalgebra c on X into a deterministic automaton on TX.

- *the trace map $\mathrm{tr}(c)\colon X \to T(B)^{A^\star}$ of such a coalgebra c is obtained via the unique coalgebra map $T(X) \to T(B)^{A^\star}$ that arises by finality in $\mathcal{EM}(T)$ in:*

$$\begin{array}{ccccc} T(B) \times T(X)^A = \widehat{M}(TX) & \dashrightarrow & \widehat{M}\big(T(B)^{A^\star}\big) = T(B) \times \big(T(B)^{A^\star}\big)^A \\ {\scriptstyle \widehat{E}(c)}\uparrow & & \cong \uparrow {\scriptstyle \zeta} \\ X \xrightarrow{\eta} TX & \dashrightarrow & T(B)^{A^\star} \\ & \searrow^{\mathrm{tr}(c)} & \end{array}$$

7.1. Non-deterministic automata, in $\mathcal{K}\ell$-style

In Subsection 5.1 we have seen how to obtain traces for non-deterministic automata via determinization (like in [36]). Now we re-describe the same example in $\mathcal{K}\ell$-style, via the isomorphisms (2). In essence we translate the "Kleisli" trace semantics approach of [18] into the current setting, like in Proposition 5. Thus we start with a non-deterministic automaton as a coalgebra of the form $c\colon X \to \mathcal{P}(1 + A \times X)$, for a fixed set of labels A and $1 = \{*\}$ modeling termination. A state $x \in X$ of such a coalgebra is final if and only if $* \in c(x)$. In this case the monad is the powerset monad $\mathcal{P}$ and the functor is $F = 1 + A \times (-)$ with finite lists $A^\star$ as initial algebra. We recall that the Kleisli category $\mathcal{K}\ell(\mathcal{P})$ is the category $\mathbf{Rel}$ of sets and relations between them, and the category $\mathcal{EM}(\mathcal{P})$ is the category $\mathbf{CL}$ of complete lattices with join-preserving maps.

The functor F lifts to $\widehat{F}\colon \mathbf{Rel} \to \mathbf{Rel}$ by Lemma 2. We instantiate M for the set $B = 1$ and the powerset monad, and get $M = 2 \times (-)^A$ since $\mathcal{P}(1) \cong 2$. Then there is an extension law $\mathfrak{e}\colon \mathcal{P}(1 + A \times (-)) \Rightarrow 2 \times \mathcal{P}^A$ with

$$\mathfrak{e}(U) \;=\; \langle o(U),\, \lambda a.\, \{x \mid \langle a, x\rangle \in U\}\rangle \qquad \text{where} \qquad o(U) \;=\; \begin{cases} 1 & \text{if } * \in U \\ 0 & \text{if } * \notin U. \end{cases}$$

One can easily check that $\mathfrak{e}$ is injective (it is actually an isomorphism) and that it satisfies the requirements (13) for an extension law.

Given a coalgebra $c\colon X \to \mathcal{P}(1 + A \times X)$ its determinization is the deterministic automaton $\widehat{E}(c)\colon \mathcal{P}(X) \to 2 \times \mathcal{P}(X)^A$ with subsets $V \subseteq X$ as states, given by $\widehat{E}(c) = \mathfrak{e} \circ \mu \circ \mathcal{P}(c)$ or, more concretely,

$$\widehat{E}(c)(V) \;=\; \left\langle o\left(\bigcup_{x\in V} c(x)\right),\ \lambda a.\ \bigcup_{x\in V} \{y \mid \langle a, y\rangle \in c(x)\} \right\rangle.$$

This determinization coincides with the well-known subset construction [20]. The trace map $\mathrm{tr}(c)$ associates with each state of the original non-deterministic automaton the language it recognizes.

The dashed map $TW \dashrightarrow Z$ in (16) in Proposition 5 is the obvious isomorphism $\mathcal{P}(A^\star) \xrightarrow{\cong} 2^{A^\star}$ for non-deterministic automata. Therefore, Proposition 5 yields that "Kleisli" trace and trace via $\mathcal{K}\ell$-extension semantics coincide, i.e. $\mathrm{tr}(c) = \mathrm{tr}_{\mathcal{K}\ell}(c)$ for any non-deterministic automaton $c\colon X \to \mathcal{P}(1 + A \times X)$.

7.2. Generative probabilistic systems

Next, we consider generative probabilistic systems with explicit termination. They also fit in the "Kleisli" trace approach of [18]. Their determinization was introduced by the last two authors of the present paper in [38] and motivated us to look at the framework of the present paper.

Generative systems are coalgebras for the functor $\mathcal{D}(1 + A \times (-))$ where $\mathcal{D}$ is the subprobability distribution monad. The functor $F = 1 + A \times (-)$ lifts to $\widehat{F}\colon \mathcal{K}\ell(\mathcal{D}) \to \mathcal{K}\ell(\mathcal{D})$ by Lemma 2. The category $\mathcal{EM}(\mathcal{D})$ is the category $\mathbf{PCA}$ of positive convex algebras and convex maps [16]. We instantiate the functor M to $B = 1$ and the subprobability distribution monad $\mathcal{D}$, and get $M = [0,1] \times (-)^A$ since $\mathcal{D}(1) \cong [0,1]$. Define $\mathfrak{e}\colon \mathcal{D}(1 + A \times (-)) \Rightarrow [0,1] \times \mathcal{D}^A$ by

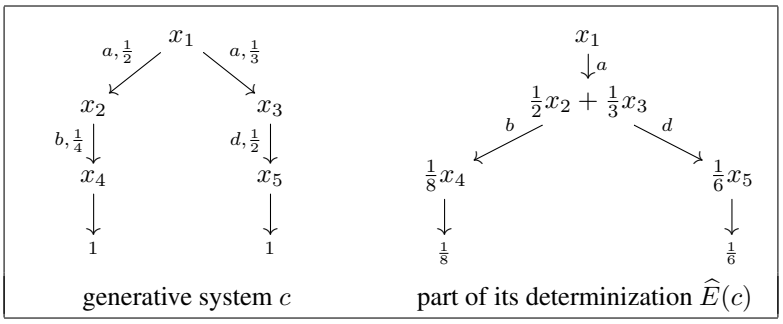
$$\mathfrak{e}(\xi) \;=\; \langle\, \xi(*),\ \lambda a.\, \lambda y.\, \xi(a, y)\,\rangle.$$

It is not difficult to check that $\mathfrak{e}$ is an extension law and that it is injective.

Given a coalgebra $c\colon X \to \mathcal{D}(1 + A \times X)$ its determinization is the deterministic automaton $\widehat{E}(c)\colon \mathcal{D}(X) \to [0,1] \times \mathcal{D}(X)^A$ with states $\mathcal{D}(X)$, given by $\widehat{E}(c) = \mathfrak{e} \circ \mu \circ \mathcal{D}(c)$ or, more concretely,

$$\widehat{E}(c)(\xi) \;=\; \left\langle\, \textstyle\sum_{x\in X} \xi(x) \cdot c(x)(*),\ \lambda a.\, \lambda y.\, \sum_{x\in X} \xi(x) \cdot c(x)(a, y)\,\right\rangle.$$

We show an example of such determinization: the automaton on the right is part of the determinization of the one on the left. The full determinization is an infinite automaton. We show the accessible part when starting from the state $\eta(x_1)$, the Dirac distribution of x_1, and we denote the distributions by formal sums. We omit zero output probabilities.



The trace map $\mathrm{tr}(c)$ associates with each state of the original generative system a subprobability distribution on words, giving the probability to terminate with a given word starting from the given state. For instance, for state x_1 above, we have $\mathrm{tr}(c)(x_1) = \frac{1}{8}ab + \frac{1}{6}ad$.

The dashed map in (16) in Proposition 5 is in this case the inclusion map $\mathcal{D}(A^\star) \to [0,1]^{A^\star}$. Therefore, again, Proposition 5 yields that "Kleisli" trace and trace via extension semantics "coincide", i.e. $\mathrm{tr}(c)(x)(w) = \mathrm{tr}_{\mathcal{K}\ell}(c)(x)(w)$ for any generative system $c\colon X \to \mathcal{D}(1 + A \times X)$, any of its states $x \in X$, and any word $w \in A^\star$. Moreover, it shows that the trace map $\mathrm{tr}(c)$ is not just any map from $A^\star$ to $[0,1]$, but a subprobability distribution on words. This coincidence was shown in [38] in concrete terms.

7.3. Weighted automata

The restrictions imposed on the monad in order for the trace semantics of [18] to work rule out several interesting monads, such as the multiset monads $\mathcal{M}_S$ (including the free vector space monad when S is a field), used for coalgebraic modeling of weighted systems. In this example, we show how the new framework allows us to deal with trace semantics for weighted systems using extension semantics. We consider the same functor $F = 1 + A \times (-)$ as before, with the multiset monad $\mathcal{M}_S$ over a semiring S. Recall that it maps a set X to the set of all finitely supported maps from X to S. Having finite support is crucial for $\mathcal{M}_S$ to be a monad, and one of the reasons why this monad does not fit in the "Kleisli" traces framework of [18]. Similar to probability distributions, we denote multisets by formal sums, now with coefficients in the semiring S. The Kleisli category $\mathcal{K}\ell(\mathcal{M}_S)$ is, for instance, the category of free commutative monoids when $S = \mathbb{N}$, and the category $\mathcal{EM}(\mathcal{M}_S)$ is the category of modules over S.

Coalgebras of the functor $\mathcal{M}_S(1 + A \times (-))$ are precisely weighted automata with weights over the set S. Given a coalgebra $c\colon X \to \mathcal{M}_S(1 + A \times X)$, each state $x \in X$ has an output weight $c(x)(*) \in S$ and an a-labelled transition into state y with weight $c(x)(\langle a, y\rangle) \in S$.

We instantiate the deterministic automaton functor M with $B = 1$ and the multiset monad $\mathcal{M}_S$, and get $M = S \times (-)^A$ since $\mathcal{M}_S(1) \cong S$. Then there is an extension natural transformation $\mathfrak{e}\colon \mathcal{M}_S(1 + A \times (-)) \Rightarrow S \times \mathcal{M}_S^A$ given, as for the subdistribution monad, by

$$\mathfrak{e}(\xi) \;=\; \langle\, \xi(*),\ \lambda a.\, \lambda x.\, \xi(\langle a, x\rangle)\,\rangle.$$

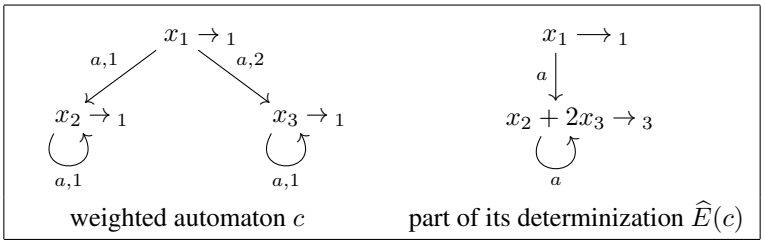
Again, $\mathfrak{e}$ satisfies all the conditions and it is injective.

For weighted automata, just like for generative systems, the determinization construction gives rise to a deterministic automaton with an infinite state-space even if the original automaton is finite. For a weighted automaton $c\colon X \to \mathcal{M}_S(1 + A \times X)$, the determinization $\widehat{E}(c)\colon \mathcal{M}_S(X) \to S \times \mathcal{M}_S(X)^A$ is given by

$$\widehat{E}(c)(\xi) \;=\; \langle\, \textstyle\sum_{x\in X} \xi(x) \cdot c(x)(*),\ \lambda a.\, \lambda y.\, \sum_{x\in X} \xi(x) \cdot c(x)(a, y)\,\rangle$$

for a multiset $\xi\colon X \to S$ with finite support $\{x \in X \mid \xi(x) \neq 0\}$. We thus get a trace map $X \to \mathcal{M}_S(X) \to S^{A^\star}$. Notice that the final coalgebra $S^{A^\star}$ in the category of $\mathcal{M}_S$-algebras can be obtained via the adjoint functors $S^{(-)}\colon \mathbf{Sets}^{\mathrm{op}} \to \mathcal{EM}(\mathcal{M}_S)$ and $\mathrm{Lin}(-, S)\colon \mathcal{EM}(\mathcal{M}_S) \to \mathbf{Sets}^{\mathrm{op}}$ from Subsection 4.1, where $\mathcal{EM}(\mathcal{M}_S)$ is the category of modules over S. Here, S plays the role of D of Subsection 4.1 and the hom-functor is now the linear mappings functor $\mathrm{Lin}(-, S)$. The endofunctors on $\mathbf{Sets}$ used for this lifting via duality are $1 + A \times (-)$ and $S \times (-)^A$.

The following is a very simple example of determinization, for $S = \mathbb{N}$.



weighted automaton c part of its determinization $\widehat{E}(c)$

In general, the transitions of the determinization of the example weighted automaton c are given by $k_1x_1 + k_2x_2 + k_3x_3 \xrightarrow{a} (k_1 + k_2)x_2 + (2k_1 + k_3)x_3$ and the termination weight of a state $k_1x_1 + k_2x_2 + k_3x_3$ equals $k_1 + k_2 + k_3$.

The trace map $\mathrm{tr}(c)\colon X \to \mathbb{N}^{A^\star}$ associates with each state x of the original weighted automaton the weighted language that it accepts. For instance, in the example above, we have $\mathrm{tr}(c)(x_1)(\langle\rangle) = 1$ and $\mathrm{tr}(c)(x_1)(a^n) = 3$ for $n \geq 1$.

Remark 3. *More specifically, we can consider weighted automata with weights over a field $\mathbb{F}$, which are coalgebras for the functor $\mathcal{M}_{\mathbb{F}}(1+A\times(-))$, where $\mathcal{M}_{\mathbb{F}}$ is the free vector space monad. This monad is special: the Kleisli category $\mathcal{K}\ell(\mathcal{M}_{\mathbb{F}})$ and the category $\mathcal{EM}(\mathcal{M}_{\mathbb{F}})$ are equivalent, since each vector space has a basis. They are the category* $\mathbf{Vec}$ *of vector spaces and linear maps over $\mathbb{F}$. The determinization $\widehat{E}(c)$ of a weighted automaton c coincides then with the* linear weighted automaton *of [8].*

In the remainder of this section we consider the quantum walks example from [21], which can be described as a coalgebra $c\colon \mathbb{Z}+\mathbb{Z} \to \mathcal{M}_{\mathbb{C}}(\mathbb{Z}+\mathbb{Z})$, where the state space $\mathbb{Z}+\mathbb{Z}$ represents positions on a line $\mathbb{Z}$, with a direction (namely $\uparrow$ for the left component in $\mathbb{Z}+\mathbb{Z}$ and $\downarrow$ for the other, downwards direction). This description only involves the (unitary) transition function given by a superposition of left and right steps. Here we adapt this example a little bit so that we can compute the resulting probabilities via traces. We take the functor $F(X) = (\mathbb{Z}+\mathbb{Z}) + X$, and coalgebra $c\colon \mathbb{Z}+\mathbb{Z} \to \mathcal{M}_{\mathbb{C}}((\mathbb{Z}+\mathbb{Z}) + (\mathbb{Z}+\mathbb{Z})) = \mathcal{M}_{\mathbb{C}}(F(\mathbb{Z}+\mathbb{Z}))$ given by:

$$\begin{array}{rcl} c(\uparrow k) & = & 1\ell(k) + \frac{1}{\sqrt{2}} \uparrow (k-1) + \frac{1}{\sqrt{2}} \downarrow (k+1) \\ c(\downarrow k) & = & 1r(k) + \frac{1}{\sqrt{2}} \uparrow (k-1) - \frac{1}{\sqrt{2}} \downarrow (k+1). \end{array}$$

The right-hand-side of these equations is a formal sum over elements of the set $F(\mathbb{Z}+\mathbb{Z}) = (\mathbb{Z}+\mathbb{Z})+(\mathbb{Z}+\mathbb{Z})$. What is possibly confusing is that in this quadruple coproduct of integers the left part $\mathbb{Z}+\mathbb{Z}$ is used for output, and the right part $\mathbb{Z}+\mathbb{Z}$ as state, for further computation. In these definitions the first parts $\ell(k)$ and $r(k)$ are the left and right part of this output. The second part involves multiplication with scalars $\pm\frac{1}{\sqrt{2}} \in \mathbb{C}$ and elements $\uparrow (k \pm 1), \downarrow (k \pm 1)$ in the right (state) component of $(\mathbb{Z}+\mathbb{Z}) + (\mathbb{Z}+\mathbb{Z})$.

As machine functor we take $M(X) = \mathcal{M}_{\mathbb{C}}(\mathbb{Z}+\mathbb{Z}) \times X$, with label set $A = 1$, and with final coalgebra $Z = \mathcal{M}_{\mathbb{C}}(\mathbb{Z}+\mathbb{Z})^{\mathbb{N}}$ of streams. The extension natural tranformation follows from additivity of the multiset monad $\mathcal{M}_{\mathbb{C}}$ (like in (2), see [12]) with κ_1 and κ_2 denoting the coproduct injections:

$$\begin{array}{ccc} \mathcal{M}_{\mathbb{C}}((\mathbb{Z}+\mathbb{Z}) + X) & \xrightarrow[\cong]{\mathfrak{e}=\lambda\varphi.\,\langle\varphi\circ\kappa_1, \varphi\circ\kappa_2\rangle} & \mathcal{M}_{\mathbb{C}}(\mathbb{Z}+\mathbb{Z}) \times \mathcal{M}_{\mathbb{C}}(X) \\ \| & & \| \\ \mathcal{M}_{\mathbb{C}}(F(X)) & & M(\mathcal{M}_{\mathbb{C}}(X)) \end{array}$$

The coalgebra $\widehat{E}(c)\colon \mathcal{M}_{\mathbb{C}}(\mathbb{Z}+\mathbb{Z}) \to \mathcal{M}_{\mathbb{C}}(\mathbb{Z}+\mathbb{Z}) \times \mathcal{M}_{\mathbb{C}}(\mathbb{Z}+\mathbb{Z})$ in the category of vector spaces over $\mathbb{C}$, resulting from Theorem 2, gives rise to a trace map $\mathrm{tr}(c)\colon \mathbb{Z}+\mathbb{Z} \to \mathcal{M}_{\mathbb{C}}(\mathbb{Z}+\mathbb{Z})^{\mathbb{N}}$. Thus, for the initial (upwards) state $\uparrow 0 \in \mathbb{Z}+\mathbb{Z}$ we get the probability $P(n,k)$ of ending up after n steps at position $k \in \mathbb{Z}$ on the line via the Born rule:

$$P(n,k) \;=\; \Big|\, \mathrm{tr}(c)(\uparrow 0)(n)(\ell k) \,\Big|^2 + \Big|\, \mathrm{tr}(c)(\uparrow 0)(n)(rk) \,\Big|^2.$$

These probabilities are computed in an ad hoc manner in [21].

8. Discussion

In this paper, we have systematically studied trace semantics, bringing together two perspectives: the coalgebraic trace semantics of [18] and the coalgebraic language equivalence via a determinization construction of [36]. Having the two perspectives together enables us to extend the class of systems that fits the framework of [18], while guaranteeing that the coalgebraic trace semantics from [18] fits in the current setting (Proposition 5). We illustrated the whole approach with several non-trivial examples, including weighted automata, quantum walks, simple Segala systems, and pushdown automata.

Our set-up has a certain overlap with [3], but the focus there is on getting coincidence of initial algebras and final coalgebras in categories $\mathcal{EM}(T)$, using stronger assumptions than here, namely commutativity of endofunctors $TF \cong GT$, see Section 6, where we only have a law $TF \Rightarrow GT$.

Acknowledgments. We are grateful to the anonymous referees for valuable comments. The work of Alexandra Silva is partially funded by the ERDF through the Programme COMPETE and by the Portuguese Foundation for Science and Technology, project ref. `FCOMP-01-0124-FEDER-020537` and `SFRH/BPD/71956/2010`.

References

[1] J.M. Autebert, J. Berstel, L. Boasson, et al. Context-free languages and pushdown automata. *Handbook of formal languages*, 1:111–174, 1997.

[2] S. Awodey. *Category Theory*. Oxford Logic Guides. OUP Oxford, 2010.

[3] A. Balan and A. Kurz. On coalgebras over algebras. *Theoretical Computer Science*, 412(38):4989–5005, 2011.

[4] M. Barr and Ch. Wells. *Toposes, Triples and Theories*. Springer, Berlin, 1985. Revized and corrected version available from URL: `www.cwru.edu/artsci/math/wells/pub/ttt.html`.

[5] N. Benton, J. Hughes, and E. Moggi. Monads and effects. In G. Barthe, P. Dybjer, L. Pinto, and J. Saraiva, editors, *APPSEM*, volume 2395 of *Lecture Notes in Computer Science*, pages 42–122. Springer, 2000.

[6] Filippo Bonchi, Marcello M. Bonsangue, Georgiana Caltais, Jan J. M. M. Rutten, and Alexandra Silva. Final semantics for decorated traces. *Electr. Notes Theor. Comput. Sci.*, 286:73–86, 2012.

[7] F. Borceux. *Handbook of Categorical Algebra*, volume 50, 51 and 52 of *Encyclopedia of Mathematics*. Cambridge Univ. Press, 1994.

[8] M. Boreale. Weighted bisimulation in linear algebraic form. In *Proc. of CONCUR '09*, pages 163–177. Springer, 2009. LNCS 5710.

[9] J.A. Brzozowski and E. Leiss. Finite automata, and sequential networks. *Theoretical Computer Science*, 10:19–35, 1980.

[10] P. Castro, P. Panangaden, and D. Precup. Equivalence relations in fully and partially observable Markov decision processes. In *Proc. IJCAI 2009*, pages 1653–1658, 2009.

[11] A.K. Chandra, D.C. Kozen, and L.J. Stockmeyer. Alternation. *Journal of the Association for Computing Machinery*, 28:114–133, 1981.

[12] D. Coumans and B. Jacobs. Scalars, monads and categories. In C. Heunen and M. Sadrzadeh, editors, *Compositional methods in Physics and Linguistics*. Oxford Univ. Press, 2012.

[13] S. Crafa and F. Ranzato. A spectrum of behavioral relations over LTSs on probability distributions. In *Proc. CONCUR 2011*, pages 124–139. LNCS 6901, 2011.

[14] K. Culik, II and J. Karhumaki. Finite automata computing real functions. *SIAM Journal of Computing*, 23(4):789–814, 1994.

[15] Y. Deng, R. van Glabbeek, M. Hennessy, and C. Morgan. Characterising testing preorders for finite probabilistic processes. *Logical Methods in Computer Science*, 4(4), 2008.

[16] E. Doberkat. Eilenberg-moore algebras for stochastic relations. *Information and Computation*, 204(12):1756–1781, 2006.

[17] Z. Ésik and W. Kuich. An algebraic generalization of omega-regular languages. In *MFCS*, pages 648–659. LNCS 3153, 2004.

[18] I. Hasuo, B. Jacobs, and A. Sokolova. Generic trace theory via coinduction. *Logical Methods in Computer Science*, 3(4:11), 2007.

[19] H. Hermanns, A. Parma, R. Segala, B. Wachter, and L. Zhang. Probabilistic logical characterization. *Information and Computation*, 209(2):154–172, 2011.

[20] J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Wesley, 2006.

[21] B. Jacobs. Coalgebraic walks, in quantum and Turing computation. In M. Hofmann, editor, *FoSSaCS*, LNCS 6604, pages 12–26. Springer, 2011.

[22] B. Jacobs. *Introduction to Coalgebra. Towards Mathematics of States and Observations*. 2012. Book, in preparation; version 2 available from `www.cs.ru.nl/B.Jacobs/CLG/JacobsCoalgebraIntro.pdf`.

[23] B. Jacobs, A. Silva, and A. Sokolova. Trace semantics via determinization. In L. Schröder and D. Pattinson, editors, *Coalgebraic Methods in Computer Science (CMCS 2012)*, LNCS 7399, pages 109–129. Springer, 2012.

[24] B. Jacobs and A. Sokolova. Exemplaric expressivity of modal logics. *Journ. of Logic and Computation*, 20(5):1041–1068, 2010.

[25] P.T. Johnstone. Adjoint lifting theorems for categories of algebras. *Bull. London Math. Soc.*, 7:294–297, 1975.

[26] Ch. Kissig and A. Kurz. Generic trace logics, 2011. `arxiv.org/abs/1103.3239`.

[27] A. Kock. Strong functors and monoidal monads. *Archiv der Mathematik*, 23, 1972.

[28] E.G. Manes. *Algebraic Theories*. Springer, Berlin, 1974.

[29] S. Mac Lane. *Categories for the Working Mathematician*. Springer, Berlin, 1971.

[30] Stefan Milius, Thorsten Palm, and Daniel Schwencke. Complete iterativity for algebras with effects. In Alexander Kurz, Marina Lenisa, and Andrzej Tarlecki, editors, *CALCO*, volume 5728 of *Lecture Notes in Computer Science*, pages 34–48. Springer, 2009.

[31] E. Moggi. Notions of computation and monads. *Information and Computation*, 93(1):55–92, 1991.

[32] Philip S. Mulry. Lifting theorems for kleisli categories. In Stephen D. Brookes, Michael G. Main, Austin Melton, Michael W. Mislove, and David A. Schmidt, editors, *MFPS*, volume 802 of *Lecture Notes in Computer Science*, pages 304–319. Springer, 1993.

[33] A. Parma and R. Segala. Logical characterizations of bisimulations for discrete probabilistic systems. In *Proc. FoSSaCS 2007*, pages 287–301. LNCS 4423, 2007.

[34] D. Pavlović, M. Mislove, and J. Worrell. Testing semantics: Connecting processes and process logics. In M. Johnson and V. Vene, editors, *Algebraic Methods and Software Technology*, LNCS 4019, pages 308–322. Springer, 2006.

[35] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. In *Proc. Concur'94*, pages 481–496. LNCS 836, 1994.

[36] A. Silva, F. Bonchi, M. Bonsangue, and J. Rutten. Generalizing the powerset construction, coalgebraically. In *Proc. FSTTCS 2010*, volume 8 of *LIPIcs*, pages 272–283, 2010.

[37] A. Silva, F. Bonchi, M. Bonsangue, and J. Rutten. Generalizing determinization from automata to coalgebras. *Logical Methods in Computer Science*, 9(1), 2013.

[38] A. Silva and A. Sokolova. Sound and complete axiomatization of trace semantics for probabilistic systems. In *Proc. MFPS 2011*, pages 291–311. ENTCS 276, 2011.

[39] D. Varacca and G. Winskel. Distributing probability over non-determinism. *Mathematical Structures in Computer Science*, 16(1):87–113, 2006.