Distributive Laws and Decidable Properties of SOS Specifications*

Bartek Klin
University of Warsaw
klin@mimuw.edu.pl

Beata Nachyła
Institute of Computer Science, Polish Academy of Sciences
beatanachyla@gmail.com

Some formats of well-behaved operational specifications, correspond to natural transformations of certain types (for example, GSOS and coGSOS laws). These transformations have a common generalization: distributive laws of monads over comonads. We prove that this elegant theoretical generalization has limited practical benefits: it does not translate to any concrete rule format that would be complete for specifications that contain both GSOS and coGSOS rules. This is shown for the case of labeled transition systems and deterministic stream systems.

1 Introduction

Distributive laws (see [14, 8] for more information) are an abstract approach to several kinds of well-behaved operational specifications. For example, for a fixed set A of labels, a family of inference rules

$$\frac{x \xrightarrow{a} x' \quad y \xrightarrow{a} y'}{x \otimes y \xrightarrow{a} x' \otimes y'} \qquad (\text{for } a \in A)$$

that define synchronous composition over labeled transition systems (LTSs), can be presented as a natural transformation $\lambda : \Sigma B \Longrightarrow B\Sigma$ (a distributive law of Σ over B), where $\Sigma X = X \times X$ and $BX = \mathscr{P}_\omega(A \times X)$ are functors on the category **Set** of sets and functions. Similarly, a family of rules

$$\frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} y'}{x \rtimes y \xrightarrow{a} y' \rtimes x'} \qquad (\text{for } a, b \in A)$$

that define an alternating composition operator $\rtimes$ on infinite streams of labels, can be understood as a transformation $\lambda : \Sigma B \Longrightarrow B\Sigma$ where $\Sigma X = X \times X$ again, and $BX = A \times X$.

Typically Σ is a polynomial functor arising from an algebraic signature. Specifications that give rise to distributive laws of Σ over B enjoy several desirable properties: they induce a B-coalgebra (e.g. an LTS) on the carrier of the initial Σ-algebra (the algebra of Σ-terms) so that bisimilarity is a congruence, and they provide an interpretation of the signature on the final B-coalgebra (provided that it exists).

These desirable properties extend to other, more expressive types of laws, including:

(a) *GSOS laws* $\rho : \Sigma(B \times \mathrm{Id}) \Longrightarrow B\Sigma^*$, where Σ^* is the free monad over Σ (see Section 2.1),

(b) *coGSOS laws* $\rho : \Sigma B^\infty \Longrightarrow B(\mathrm{Id} + \Sigma)$, where B^∞ is the cofree comonad over B (see Section 2.2),

(c) *distributive laws of monads over comonads*, i.e., natural transformations $\lambda : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$ subject to a few axioms. (In this paper we only consider distributive laws of free monads over cofree comonads, see Section 2.4.)

*This work was supported by the Polish National Science Centre (NCN) grant 2012/07/E/ST6/03026.

J. Borgström, S. Crafa (Eds.): Combined Workshop on Expressiveness in Concurrency and Structural Operational Semantics (EXPRESS/SOS 2014)
EPTCS 160, 2014, pp. 79–93, doi:10.4204/EPTCS.160.8

GSOS and coGSOS laws are incomparable, i.e., there are specifications that conform to one type but not the other, and distributive laws of monads over comonads are a common generalization of both. From now on, for brevity, we shall call them simply *distributive laws*.

For standard examples of B, GSOS and coGSOS laws correspond to *rule formats*, i.e., syntactic restrictions on the form of inference rules that are allowed in a specification for it to define a corresponding type of law. For $BX = \mathscr{P}_\omega(A \times X)$, where $\mathscr{P}_\omega$ is the finite powerset functor, it was observed in [14] that GSOS laws correspond to previously known GSOS [2] specifications (hence the name of the law type), that allow rules such as:

$$\frac{x_1 \xrightarrow{a_{1,1}} y_{1,1} \quad x_1 \xrightarrow{a_{1,2}} y_{1,2} \quad \cdots \quad x_i \xrightarrow{a_{i,j}} y_{i,j} \quad \cdots \quad x_i \not\xrightarrow{b_{i,j}} \quad \cdots}{\mathtt{f}(x_1,\ldots,x_k) \xrightarrow{b} \mathtt{t}}$$

where variables x_i can be tested for the presence and/or absence of transitions labeled with different labels, and the resulting transition can go to an arbitrary term $\mathtt{t}$ built over the variables x_i and $y_{i,j}$. On the other hand, coGSOS laws for the same functor B are induced by *safe ntree* [5, 14] specifications, where additionally *lookahead* is allowed, i.e., variables that are targets of premise transitions can be further tested for other transitions as in the rule:

$$\frac{x \xrightarrow{a} y \xrightarrow{b} z}{\mathtt{f}(x) \xrightarrow{c} \mathtt{g}(z)}$$

On the other hand, coGSOS is restricted in that the target term $\mathtt{t}$ in the conclusion must be either a variable or a flat term built of a single operation symbol and variables.

Both GSOS and coGSOS laws, are generalized by distributive laws. In fact, desired properties of systems induced by GSOS and coGSOS laws were proved in [14] by showing first that these laws induce distributive laws, and then proving those properties for the latter, more general laws. This is tantalizing, as it suggests that for standard functors B one could find new, more expressive syntactic rule formats that would correspond to distributive laws and hence guarantee good properties of specifications. The problem of finding such a format was left open in [14] and mentioned as still open in later works [1, 8].

The purpose of this paper is to suggest a negative answer to that problem. Specifically, we claim that there is no rule format that would adequately recognize those specifications that induce distributive laws of monads over comonads, within a class of specifications that extends both GSOS and coGSOS.

This claim is rather vague, and we must make it precise before we attempt to prove it. First of all, there is no hope to prove it for all monads and comonads; clearly, for some trivial monads and comonads all distributive laws are easily enumerated, and even for some nontrivial comonads a complete description of distributive laws is known [7]. Therefore in this paper we shall consider laws $\lambda : \Sigma^* B^\infty \to B^\infty \Sigma^*$ for Σ^* the free monad over a polynomial functor Σ, and B^∞ the cofree comonad over $BX = A \times X$, pertaining to stream systems, or $BX = \mathscr{P}_\omega(A \times X)$, pertaining to labeled transition systems. Hopefully it shall be clear how our arguments for the lack of expressive formats for these two behaviour functors, might extend to other standard functors used to model transition systems coalgebraically.

To make our claim precise, the first question we need to answer is: *what is a format?* In positive results about GSOS and coGSOS laws mentioned above, the answer was easy: one simply formulated some "syntactic forms" of rules and provided ways of defining laws from sets of rules that conformed to them. Now that we want a negative result, we need to quantify over all "syntactic forms", so we need to understand what a syntactic form is in general. We opt for a general and permissive answer: *a format is a decidable property of specifications.* Indeed, no matter what a "format" may be, it should be effectively checkable whether a specification conforms to it.

This leads to another question: *what is a specification?* Some definitions of this term would immediately invalidate our claim; for example, if we say that "a specification is either a GSOS specification or a coGSOS specification", then every specification induces a distributive law as described already in [14] and the problem is trivially decidable. However, we are interested in more permissive notions of specification that would allow a more substantial combination of GSOS and coGSOS features. We therefore focus on *mixed-GSOS specifications*, where every rule is either a GSOS or a coGSOS rule.

Note that there are other interesting notions of specification where the claim becomes false. For example, as proved in [13], in the context of LTSs one may consider so-called (positive) *tyft/tyxt* [4] specifications, and guarantee the existence of a distributive law for every specification. However, tyft/tyxt specifications extend neither full GSOS nor coGSOS, so this does not match the abstract observation that distributive laws generalize both GSOS laws and coGSOS laws.

There is still one vague point in our claim: *what does it mean for a specification to induce a distributive law?* In positive results about GSOS and coGSOS specifications [8, 14], one simply provides particular ways of inducing distributive laws from specifications that look so natural that everybody is convinced. Here, to show undecidability, we shall need to prove that some instances do *not* induce distributive laws, so we need to quantify over all possible "ways of inducing laws", a vague notion itself.

We approach this problem by observing that every mixed-GSOS specification induces, in a very natural way, a natural transformation $\rho : \Sigma B^\infty \Longrightarrow B\Sigma^*$ which we call a *biGSOS law*. Then we define (Definition 13) what it means for a distributive law λ to *extend* a biGSOS law ρ; essentially, λ must restrict to ρ when composed with obvious inclusions and projections. Our claim then becomes:

Claim. *It is undecidable whether a given mixed-GSOS specification extends to a unique distributive law.*

One may worry whether our insistence on a unique extension is not overly restrictive. Indeed, perhaps sometimes a specification may extend to several distributive laws, but one of these laws is somehow better than the other ones, for example (in the LTS setting) the least one, or canonical in some other way? However, as will be evident from our proofs, this is not a problem: all our instances of specifications will either extend to one distributive law or to none at all, therefore no matter what notion of "canonical extension" one may come up with, the problem remains undecidable.

We prove undecidability by reduction from the halting problem of a variant of queue machines defined in Section 4. Then, in Section 5, we prove the Claim for the case of stream systems (Theorem 24), and in Section 6 we explain how the proof is adapted to the case of LTSs (Theorem 29).

2 Preliminaries

The reader should be familiar with notions of category theory such as functors and natural transformations, see e.g. [11]. All functors we consider are endofunctors on the category of sets and functions.

2.1 Algebras and monads

An *algebra* for a functor Σ is a set X (the *carrier*) together with a function $g : \Sigma X \to X$ (the *structure*). An algebra morphism from $g : \Sigma X \to X$ to $h : \Sigma Y \to Y$ is a function $f : X \to Y$ such that $f \circ g = h \circ \Sigma f$. Algebras for Σ and their morphisms form a category. Of particular interest in this category are initial objects, i.e., initial Σ-algebras.

Assume that, for any set X, an initial algebra for the functor $\Sigma(-) + X$ exists, denote its carrier $\Sigma^* X$ and its structure by:

$$\Sigma\Sigma^* X \xrightarrow{\psi_X} \Sigma^* X \xleftarrow{\eta_X} X.$$

Then Σ^*, defined on functions using initiality, becomes a functor and $\psi : \Sigma\Sigma^* \Longrightarrow \Sigma^*$ and $\eta : \mathrm{Id} \Longrightarrow \Sigma^*$ are natural transformations. Moreover, Σ^* is a *monad*, i.e., it is equipped with a natural transformation $\mu : \Sigma^*\Sigma^* \Longrightarrow \Sigma^*$ such that the following diagrams commute:

$$\begin{array}{ccccc} \Sigma^* & \overset{\Sigma^*\eta}{\Longrightarrow} & \Sigma^*\Sigma^* & \overset{\eta\Sigma^*}{\Longleftarrow} & \Sigma^* \\ & \searrow & \downarrow{\scriptstyle \mu} & \swarrow & \\ & & \Sigma^* & & \end{array} \qquad \begin{array}{ccc} \Sigma^*\Sigma^*\Sigma^* & \overset{\Sigma^*\mu}{\Longrightarrow} & \Sigma^*\Sigma^* \\ {\scriptstyle \mu\Sigma^*}\downarrow & & \downarrow{\scriptstyle \mu} \\ \Sigma^*\Sigma^* & \underset{\mu}{\Longrightarrow} & \Sigma^*. \end{array} \tag{1}$$

Σ^* is called the *free monad* over Σ. Another relevant transformation is $\iota : \Sigma \Longrightarrow \Sigma^*$ defined by $\iota = \psi \circ \Sigma\eta$; it further satisfies the equation $\psi = \mu \circ \iota\Sigma^*$.

Example 1 Any algebraic signature $(\mathsf{q}_i)_{i\in I}$, where each q_i is an operation symbol of arity $n_i \in \mathbb{N}$, gives rise to an endofunctor $\Sigma X = \coprod_{i\in I} X^{n_i}$. Then Σ-algebras are algebras for the signature in the sense of universal algebra, and Σ-algebra morphisms are exactly algebra homomorphisms. Moreover, $\Sigma^* X$ is the set of terms over the signature with variables taken from X, η interprets variables as terms, ψ and μ glue together terms built of terms, and ι interprets terms built of single operation symbols as terms.

2.2 Coalgebras and comonads

The following development is dual to the one for algebras and monads; we include it for completeness and to introduce some basic terminology and notation. For more information about coalgebras, see [12].

A *coalgebra* for a functor B is a set X (the *carrier*) together with a function $g : X \to BX$ (the *structure*). A coalgebra morphism from $g : X \to BX$ to $h : Y \to BY$ is a function $f : X \to Y$ such that $h \circ f = Bf \circ g$. Coalgebras for B and their morphisms form a category.

Assume that, for any set X, a final coalgebra for the functor $B(-) \times X$ exists, denote its carrier $B^\infty X$ and its structure by:

$$BB^\infty X \xleftarrow{\theta_X} B^\infty X \xrightarrow{\varepsilon_X} X.$$

Then B^∞, defined on functions using finality, becomes a functor and $\theta : B^\infty \Longrightarrow BB^\infty$ and $\varepsilon : B^\infty \Longrightarrow \mathrm{Id}$ are natural transformations. Moreover, B^∞ is a *comonad*, i.e., it is equipped with a natural transformation $\delta : B^\infty \Longrightarrow B^\infty B^\infty$ such that diagrams dual to (1) commute. B^∞ is called the *cofree comonad* over B. Another relevant transformation is $\pi : B^\infty \Longrightarrow B$ defined by $\pi = B\varepsilon \circ \theta$; it further satisfies the equation $\theta = \pi B^\infty \circ \delta$.

Example 2 Let $BX = A \times X$, for a fixed set A of *labels*. B-coalgebras are *stream systems*, i.e., sets X (of *states*) equipped with functions to A and to X again; the intuition is that a state produces a label and transforms into another state. The cofree comonad over B is given by $B^\infty X = (X \times A)^\omega$; we will depict elements of $B^\infty X$ as streams of labeled transitions:

$$B^\infty X \ni \sigma = x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \xrightarrow{a_3} \cdots$$

with $x_i \in X$ and $a_i \in A$. For any $n \in \mathbb{N}$, by $\sigma^{(n)} \in B^\infty X$ denote the n-th tail of σ, i.e., the substream of σ that starts at x_n. Natural transformations explained above are then given by:

$$\varepsilon_X(\sigma) = x_0 \qquad \theta_X(\sigma) = \big(a_0, \sigma^{(1)}\big)$$
$$\delta_X(\sigma) = \big(\sigma \xrightarrow{a_0} \sigma^{(1)} \xrightarrow{a_1} \sigma^{(2)} \xrightarrow{a_2} \sigma^{(3)} \longrightarrow \cdots\big) \qquad \pi_X(\sigma) = (a_0, x_1)$$

One may look at elements of $B^\infty X$ as streams of labels "colored" with elements of X; elements of $B^\infty B^\infty X$ are then streams colored by streams, and $\delta_X(\sigma)$ is the stream that arises from σ by coloring each node with the substream of σ that starts in it.

Example 3 Let $\mathscr{P}_\omega$ denote the finite powerset functor, and let $BX = \mathscr{P}_\omega(A \times X)$, for a fixed set A of labels. B-coalgebras are *(finitely branching) labeled transition systems*. The cofree comonad over B is a functor B^∞ that maps a set X to the set of finitely branching, but possibly infinitely deep trees, edge-labeled with elements of A and node-colored by elements of X, quotiented by a version of strong bisimilarity that takes into account both edge labels and node colors.

Natural transformations listed above are defined by analogy to Example 2. For a tree $\mathtt{T} \in B^\infty X$:

- $\varepsilon_X(\mathtt{T}) \in X$ is the color of the root node of $\mathtt{T}$,
- $\delta_X(\mathtt{T}) \in B^\infty B^\infty X$ arises from $\mathtt{T}$ by coloring every node with the subtree rooted in it,
- $\theta_X(\mathtt{T}) \in BB^\infty X$ is the set of immediate subtrees of the root together with labels of the edges that lead to these subtrees,
- $\pi_X(\mathtt{T}) \in BX$ is similar, but with the immediate subtrees replaced by the colors of their roots.

A little care is needed to show that components of these transformations are well-defined on bisimilarity classes of trees. For example, if $\mathtt{T}_1$ and $\mathtt{T}_2$ are related by a bisimulation, then $\delta_X(\mathtt{T}_1)$ and $\delta_X(\mathtt{T}_2)$ also are, as bisimilar nodes get assigned the same colors (here colors are bisimilarity classes of trees).

2.3 GSOS and coGSOS laws

Algebras, coalgebras, monads and comonads can be combined in distributive laws of various kinds. We only recall a few basic definitions and examples here; for a more comprehensive treatment see [8].

For any functor B, denote $\tilde{B} = \mathrm{Id} \times B$.

Definition 4 Given endofunctors Σ and B such that the free monad Σ^* over Σ exists, a *GSOS law* is a natural transformation $\rho : \Sigma\tilde{B} \Longrightarrow B\Sigma^*$.

Example 5 Consider $BX = A \times X$, and let $\Sigma X = X \times X$ arise from a signature with a single binary function symbol $\mathtt{zip}$. A family of rules

$$\frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} y'}{\mathtt{zip}(x,y) \xrightarrow{a} \mathtt{zip}(y,x')} \qquad (\text{for } a,b \in A)$$

together defines a GSOS law by:

$$\rho_X(\mathtt{zip}((x,(a,x')),(y,(b,y')))) = (a,\mathtt{zip}(y,x'))$$

for any $x,x',y,y' \in X$ and $a,b \in A$.

Dually, for any functor Σ, denote $\bar{\Sigma} = \mathrm{Id} + \Sigma$.

Definition 6 Given endofunctors Σ and B such that the cofree comonad B^∞ over B exists, a *coGSOS law* is a natural transformation $\rho : \Sigma B^\infty \Longrightarrow B\bar{\Sigma}$.

Example 7 Consider $BX = A \times X$, and let $\Sigma X = X$ arise from a signature with a single unary function symbol $\mathtt{q}$. The family of rules (that define a unary operation that drops every second label from a given stream):

$$\frac{x \xrightarrow{a_1} x' \xrightarrow{a_2} x''}{\mathtt{q}(x) \xrightarrow{a_2} \mathtt{q}(x'')} \text{ (for } a_1, a_2 \in A)$$

together defines a coGSOS law by:

$$\rho_X(\mathtt{q}(x \xrightarrow{a_1} x' \xrightarrow{a_2} x'' \xrightarrow{a_3} \cdots)) = (a_2, \mathtt{q}(x''))$$

for any $x, x', x'', \ldots \in X$ and $a_1, a_2, a_3, \ldots \in A$.

Example 8 Now, consider the LTS behaviour functor $BX = \mathscr{P}_\omega(A \times X)$, and let $\Sigma X = X$ as in Example 7. The rules:

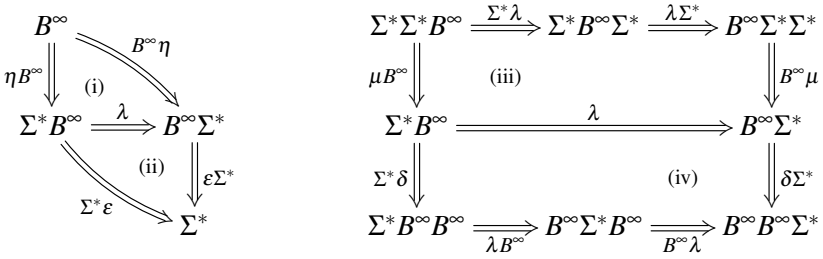
$$\frac{x \xrightarrow{a_1} x' \xrightarrow{a_2} x''}{\mathtt{q}(x) \xrightarrow{a_2} \mathtt{q}(x'')} \qquad \frac{x \xrightarrow{a_1} x' \not\longrightarrow}{\mathtt{q}(x) \xrightarrow{a_1} \mathtt{q}(x)} \qquad (\text{for } a_1, a_2 \in A)$$

define a coGSOS law $\rho : \Sigma B^\infty \Longrightarrow B\bar{\Sigma}$, where $\rho_X(\mathtt{q}(\mathtt{T}))$ is the set of pairs $(a, \mathtt{q}(x))$ such that $\mathtt{T}$ has (1) a two-step path from the root to a node colored by x with the second step labeled by a, or (2) a single step, labeled with a, to a leaf (i.e. a node without successors), and the root of $\mathtt{T}$ is colored by x.

2.4 Distributive laws

In [14] it was noticed that both GSOS and coGSOS laws are generalized by distributive laws of monads over comonads; in this paper we call them simply *distributive laws*.

Definition 9 A distributive law of a monad (Σ^*, η, μ) over a comonad $(B^\infty, \varepsilon, \delta)$ is a natural transformation $\lambda : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$ subject to the following four axioms:



Example 10 Consider $BX = A \times X$ as in Example 2, and a functor $\Sigma X = \coprod_{q \in Q} X \cong Q \times X$ arising as in Example 1 from an algebraic signature consisting of a set of unary operation symbols $Q = \{\mathtt{q}_1, \ldots, \mathtt{q}_k\}$. Then $B^\infty X = (X \times A)^\omega$ and $\Sigma^* X = Q^* \times X$; for $\mathtt{t} \in Q^*$, we shall write $\mathtt{t}(x)$ instead of $(\mathtt{t}, x) \in \Sigma^* X$, and simply x instead of $\varepsilon(x)$, for the empty string $\varepsilon \in Q^*$.

For a distributive law $\lambda : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$, the naturality condition means that if

$$\lambda_X(\mathtt{t}(x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} \cdots)) = \tau_0 \xrightarrow{b_0} \tau_1 \xrightarrow{b_1} \tau_2 \xrightarrow{b_2} \cdots$$
$$\lambda_Y(\mathtt{t}(y_0 \xrightarrow{a_0} y_1 \xrightarrow{a_1} y_2 \xrightarrow{a_2} \cdots)) = \gamma_0 \xrightarrow{c_0} \gamma_1 \xrightarrow{c_1} \gamma_2 \xrightarrow{c_2} \cdots$$

then for all $i \in \mathbb{N}$ one has $b_i = c_i$, and $\gamma_i \in \Sigma^* Y$ arises from $\tau_i \in \Sigma^* X$ by substituting each x_j by the corresponding y_j. Informally, the value of λ_X on $\mathtt{t}(\sigma)$ essentially depends only on the term $\mathtt{t}$ and on the labels in the stream σ, and the colors x_j in σ are merely rearranged into terms τ_k independently from their identity or structure. This also implies that all elements from X present in $\lambda_X(\mathtt{t}(\sigma))$ must have been present in σ.

Further, the four axioms of Definition 9 amount to:

(i) $\lambda_X(x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} \cdots) = x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} \cdots$,

(ii) if $\lambda_X(\mathtt{t}(x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} \cdots)) = \tau_0 \xrightarrow{b_0} \tau_1 \xrightarrow{b_1} \tau_2 \xrightarrow{b_2} \cdots$ then $\tau_0 = \mathtt{t}(x_0)$,

(iii) if $\lambda_X(\mathtt{s}(x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} \cdots)) = \tau_0 \xrightarrow{b_0} \tau_1 \xrightarrow{b_1} \tau_2 \xrightarrow{b_2} \cdots$ and
$\lambda_{\Sigma^* X}(\mathtt{t}(\tau_0 \xrightarrow{b_0} \tau_1 \xrightarrow{b_1} \tau_2 \xrightarrow{b_2} \cdots)) = \gamma_0 \xrightarrow{c_0} \gamma_1 \xrightarrow{c_1} \gamma_2 \xrightarrow{c_2} \cdots$ then
$\lambda_X(\mathtt{ts}(x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} \cdots)) = \gamma_0 \xrightarrow{c_0} \gamma_1 \xrightarrow{c_1} \gamma_2 \xrightarrow{c_2} \cdots$.

Informally, λ is defined compositionally with respect to Σ-terms.

(iv) if $\lambda_X(\mathtt{t}(x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} \cdots)) = \tau_0 \xrightarrow{b_0} \tau_1 \xrightarrow{b_1} \tau_2 \xrightarrow{b_2} \cdots$ then for every $i \in \mathbb{N}$,
$\lambda_X(\overline{\tau_i}) = \tau_i \xrightarrow{b_i} \tau_{i+1} \xrightarrow{b_{i+1}} \cdots$, where $\overline{\tau_i} \in \Sigma^* B^\infty X$ arises from $\tau_i \in \Sigma^* X$ by replacing every x_j with the stream starting at it. Informally, λ_X is defined "decompositionally" with respect to streams.

3 BiGSOS laws and mixed-GSOS specifications

In [14] it was proved that (1) every GSOS law induces a distributive law and (2) every coGSOS law induces a distributive law. The two ways of inducing distributive laws explained there are both natural and convincing, but formally different. We wish to study the problem of inducing distributive laws from specifications that would generalize both GSOS and coGSOS laws, so we need to have a general understanding of what it means to induce a distributive law. To this end, we consider the following simple generalization of GSOS and coGSOS:

Definition 11 Given endofunctors Σ and B such that the free monad Σ^* over Σ exists and the cofree comonad B^∞ over B exist, a *biGSOS law* is a natural transformation $\rho : \Sigma B^\infty \Longrightarrow B\Sigma^*$.

GSOS and coGSOS laws give rise to biGSOS laws by composing with injections or projections:

$$\Sigma B^\infty \xRightarrow{\Sigma\langle\varepsilon,\pi\rangle} \Sigma\tilde{B} \xRightarrow{\rho'} B\Sigma^*, \qquad \Sigma B^\infty \xRightarrow{\rho''} B\bar{\Sigma} \xRightarrow{B[\eta,\iota]} B\Sigma^*.$$

where ρ' is a GSOS law and ρ'' is a coGSOS law. (Note that $\langle\varepsilon,\pi\rangle : B^\infty \Longrightarrow \tilde{B}$ and $[\eta,\iota] : \bar{\Sigma} \Longrightarrow \Sigma^*$ are natural transformations.) As a result, biGSOS laws generalize both GSOS and coGSOS laws. However, they offer much more flexibility. In particular, for the case of stream systems and LTSs, we consider:

Definition 12 A stream (or LTS) specification is *mixed-GSOS* if every rule in it is either a GSOS rule or a coGSOS rule, and moreover, for any operator $\mathtt{f}$, rules that define $\mathtt{f}$ (i.e., those that have $\mathtt{f}$ on the left side of the conclusion) are either all GSOS or all coGSOS.

Note that we allow coGSOS-defined operations in conclusions of GSOS rules (and vice versa), so that e.g. the specification in Example 14 below is mixed-GSOS.

One could also define mixed GSOS more abstractly, by partitioning the signature into two disjoint subsignatures, $\Sigma = \Sigma_{\mathrm{GSOS}} + \Sigma_{\mathrm{coGSOS}}$, and requesting two natural transformations:

$$\rho_{\mathrm{GSOS}} : \Sigma_{\mathrm{GSOS}}\tilde{B} \Longrightarrow B\Sigma^* \qquad \rho_{\mathrm{coGSOS}} : \Sigma_{\mathrm{coGSOS}} B^\infty \Longrightarrow B\bar{\Sigma},$$

one responsible for the GSOS, the other one the coGSOS part of the specification. It is then clear how a mixed-GSOS specification induces a biGSOS law, by comparing ρ_{GSOS} and ρ_{coGSOS} composed with suitable injections and projections. Note that biGSOS laws allow still more flexibility than allowed by mixed-GSOS, as they allow rules that combine complex conclusion terms as in GSOS, with lookahead as in coGSOS.

It may not be evident what it means for a biGSOS law to induce a distributive law, but it is clear how a given distributive law may extend a biGSOS law, by composing with relevant injections and projections:

Definition 13 A distributive law $\lambda : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$ *extends* a biGSOS law $\rho : \Sigma B^\infty \Longrightarrow B\Sigma^*$ if the following diagram commutes:

$$\begin{array}{ccc} \Sigma B^\infty & \xRightarrow{\rho} & B\Sigma^* \\ {\scriptstyle \iota B^\infty}\Downarrow & & \Uparrow{\scriptstyle \pi\Sigma^*} \\ \Sigma^* B^\infty & \xRightarrow[\lambda]{} & B^\infty \Sigma^* \end{array} \tag{2}$$

In other words, λ extends ρ if it equals ρ when its arguments are restricted to Σ-terms of depth 1 and results projected to B-behaviours of depth 1.

As the following examples show, not every biGSOS law extends to a distributive law, and those that do may not extend uniquely.

Example 14 For $BX = A \times X$ with a chosen element $\$ \in A$, consider syntax with one constant $\mathtt{C}$ and one unary operation $\mathtt{q}$, so that $\Sigma X = 1 + X$ and $B^\infty X = (X \times A)^\omega$. Consider $\rho : \Sigma B^\infty \Longrightarrow B\Sigma^*$ defined by rules:

$$\frac{}{\mathtt{C} \xrightarrow{\$} \mathtt{q}(\mathtt{C})} \qquad \frac{x \xrightarrow{a} x' \xrightarrow{b} x''}{\mathtt{q}(x) \xrightarrow{b} \mathtt{q}(x'')} \text{ (for } a, b \in A)$$

Consider any distributive law $\lambda : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$, and present $\lambda_0(\mathtt{C})$ as:

$$\lambda_0(\mathtt{C}) \quad = \quad \mathtt{C} \xrightarrow{a_0} \tau_1 \xrightarrow{a_1} \tau_2 \xrightarrow{a_2} \cdots \quad \in \quad B^\infty \Sigma^* 0 \tag{3}$$

with each $\tau_i \in \Sigma^* 0$ and $a_i \in A$.

If λ extends ρ then, by (2) applied to $\mathtt{C} \in \Sigma B^\infty 0$, we have $a_0 = \$$ and $\tau_1 = \mathtt{q}(\mathtt{C})$. Since λ is a distributive law, by axioms (ii) and (iv) of Definition 9 as explained in Example 10, from (3) we get

$$\lambda_0(\mathtt{q}(\mathtt{C})) \quad = \quad \mathtt{q}(\mathtt{C}) \xrightarrow{a_1} \tau_2 \xrightarrow{a_2} \cdots \tag{4}$$

Now, by (2) applied to $\mathtt{q}(\lambda_0(\mathtt{C})) \in \Sigma B^\infty \Sigma^* 0$, we have:

$$\lambda_{\Sigma^* 0}(\mathtt{q}(\lambda_0(\mathtt{C}))) = \lambda_{\Sigma^* 0}(\mathtt{q}(\mathtt{C} \xrightarrow{a_0} \tau_1 \xrightarrow{a_1} \tau_2 \xrightarrow{a_2} \cdots)) = \mathtt{q}(\mathtt{C}) \xrightarrow{a_1} \mathtt{q}(\tau_2) \longrightarrow \cdots \tag{5}$$

(only the first step of the stream on the right is determined this way). By axiom (iii) of Definition 9 as explained in Example 10, the stream (5) is equal to (4) (or, more precisely, it is mapped to it by pointwise application of μ_0); as a result, $\tau_2 = \mathtt{q}(\tau_2)$. However, there is no such term τ_2 and, as a consequence, a distributive law λ that extends ρ does not exist.

Example 15 Consider the previous example with the rightmost rule slightly modified to:

$$\frac{x \xrightarrow{a} x' \xrightarrow{b} x''}{\mathtt{q}(x) \xrightarrow{b} x''} \text{ (for } a, b \in A)$$

If, say, $A = \{\$, €\}$, then the corresponding ρ can be extended e.g. to distributive laws λ, λ' such that:

$$\lambda_0(\mathtt{C}) = \mathtt{C} \xrightarrow{\$} \mathtt{q}(\mathtt{C}) \xrightarrow{\$} \mathtt{q}(\mathtt{C}) \xrightarrow{\$} \mathtt{q}(\mathtt{C}) \xrightarrow{\$} \cdots$$
$$\lambda'_0(\mathtt{C}) = \mathtt{C} \xrightarrow{\$} \mathtt{q}(\mathtt{C}) \xrightarrow{€} \mathtt{q}(\mathtt{C}) \xrightarrow{€} \mathtt{q}(\mathtt{C}) \xrightarrow{€} \cdots$$

This example shows that distinct distributive laws $\lambda, \lambda' : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$ can sometimes be equalized by composing with both $\iota B^\infty : \Sigma B^\infty \Longrightarrow \Sigma^* B^\infty$ and $\pi \Sigma^* : B^\infty \Sigma^* \Longrightarrow B\Sigma^*$ (see Definition 13). However, distinct distributive laws cannot be equalized by composing with only one of these transformations:

Lemma 16 For any distributive laws $\lambda, \lambda' : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$:

(a) if $\lambda \circ \iota B^\infty = \lambda' \circ \iota B^\infty$ then $\lambda = \lambda'$, and (b) if $\pi\Sigma^* \circ \lambda = \pi\Sigma^* \circ \lambda'$ then $\lambda = \lambda'$.

It makes sense to say that a biGSOS law ρ induces a distributive law if there is a unique distributive law that extends ρ. This is consistent with known results about GSOS and coGSOS laws, which, as has been understood since [14], induce distributive laws:

Theorem 17 For every GSOS law $\rho : \Sigma \tilde{B} \Longrightarrow B\Sigma^*$, and for every coGSOS law $\rho : \Sigma B^\infty \Longrightarrow B\bar{\Sigma}$ there is a unique distributive law $\lambda : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$ that extends the associated biGSOS law.

Proof sketch. For the existence of λ, constructions of distributive laws from GSOS and coGSOS laws were given already in [14], and later explained more elegantly in [10]. It is not difficult to prove that those constructions extend the respective GSOS and coGSOS laws in the sense of Definition 13.

For the uniqueness of λ, Lemma 16 is used. □

4 Queue machines

We shall prove that it is undecidable whether a given biGSOS law uniquely extends to a distributive law. To this end, we use the undecidability of the halting problem of queue machines.

A queue machine (QM) is a deterministic finite automaton additionally equipped with a first-in-first-out queue to store letters. A machine can read letters off the queue, and depending on their contents change their state while adding new letters to the queue. Under the classical definition [9], a QM in each transition (a) removes exactly one letter from the queue and (b) adds some (possibly zero) letters to it. For our purposes, it will be convenient to consider instead a variant of QMs that, in each step: (a) remove zero, one or two letters from the queue, and (b) add exactly one letter to it. Formally:

Definition 18 A *queue machine* (QM) $\mathscr{M} = (Q, A, \$, q_1, \delta_0, \delta_1, \delta_2)$ consists of a finite set Q of states, a finite alphabet A with a chosen symbol $\$ \in A$, a starting state $q_1 \in Q$, and three partial transition functions:

$$\delta_0 : Q \rightharpoonup Q \times A \qquad \delta_1 : Q \times A \rightharpoonup Q \times A \qquad \delta_2 : Q \times A \times A \rightharpoonup Q \times A$$

that are disjointly defined and jointly total, i.e., such that for each $q \in Q$ and $a, b \in A$, exactly one of $\delta_0(q)$, $\delta_1(q,a)$ or $\delta_2(q,a,b)$ is defined. A *configuration* of $\mathscr{M}$ is a pair $(q,w) \in Q \times A^*$; the machine induces a transition function $\longrightarrow\!\!\triangleright$ on the set of configurations by:

$$\begin{aligned}
(q,w) &\longrightarrow\!\!\triangleright (q', wc) && \text{if} && \delta_0(q) = (q', c)\\
(q,aw) &\longrightarrow\!\!\triangleright (q', wc) && \text{if} && \delta_0(q) \text{ undefined and } \delta_1(q,a) = (q', c)\\
(q,abw) &\longrightarrow\!\!\triangleright (q', wc) && \text{if} && \delta_0(q) \text{ and } \delta_1(q,a) \text{ undefined and } \delta_2(q,a,b) = (q', c).
\end{aligned}$$

Note that an MQM never makes a queue empty, and it terminates if and only if it reaches a configuration (q,a) with a single letter a in the queue, such that $\delta_0(q)$ and $\delta_1(q,a)$ are undefined.

Theorem 19 It is undecidable whether a given QM terminates from the configuration $(q_1,\$)$, called the *initial configuration*.

Proof. As is well known, it is undecidable whether a classical QM $\mathscr{M}$ as defined in [9] terminates on its initial configuration. For every classical $\mathscr{M}$ one constructs a QM $\overline{\mathscr{M}}$ as in Definition 18 that terminates on its initial configuration if and only if $\mathscr{M}$ does. □

5 From queue machines to stream specifications

Given a QM $\mathscr{M} = (Q,A,\$,q_1,\delta_0,\delta_1,\delta_2)$, consider a signature with a single constant $\mathtt{C}$ and a family of unary operation symbols $\{\mathtt{q} \mid q \in Q\}$, and a family of rules:

$$\frac{}{\mathtt{C} \xrightarrow{\$} \mathtt{q}_1(\mathtt{C})}\ (\mathbf{C}) \qquad \frac{}{\mathtt{q}(x) \xrightarrow{c} \mathtt{q}'(x)}\ (\mathbf{R0}) \qquad \frac{x \xrightarrow{a} y}{\mathtt{q}(x) \xrightarrow{c} \mathtt{q}'(y)}\ (\mathbf{R1}) \qquad \frac{x \xrightarrow{a} y \xrightarrow{b} z}{\mathtt{q}(x) \xrightarrow{c} \mathtt{q}'(z)}\ (\mathbf{R2}) \tag{6}$$

for all $q,q' \in Q$ and $a,b,c \in A$ subject to the following conditions:

- **R0** is included when $\delta_0(q) = (q',c)$,
- **R1** is included when $\delta_0(q)$ is undefined and $\delta_1(q,a) = (q',c)$, and
- **R2** is included when $\delta_0(q)$ and $\delta_1(q,a)$ are undefined and $\delta_2(q,a,b) = (q',c)$.

These rules are mixed GSOS, so they define a biGSOS law $\rho_{\mathscr{M}} : \Sigma B^\infty \Longrightarrow B\Sigma^*$, where $BX = A \times X$ and $\Sigma X = 1 + Q \times X$. We shall now prove, in a sequence of lemmas, that $\rho_{\mathscr{M}}$ uniquely extends to a distributive law if and only if $\mathscr{M}$ does *not* terminate from the initial configuration. Our argument relies on the following correspondence between partial runs of $\mathscr{M}$ and prefixes of streams produced by distributive laws that extend $\rho_{\mathscr{M}}$:

Lemma 20 For every $n > 0$, if a QM $\mathscr{M}$ makes $n-1$ steps from the initial configuration:

$$q_1,w_1 \longrightarrow\!\!\triangleright\ q_2,w_2 \longrightarrow\!\!\triangleright\ q_3,w_3 \longrightarrow\!\!\triangleright\ \cdots \longrightarrow\!\!\triangleright\ q_n,w_n$$

(where $w_1 = \$$) then every distributive law λ that extends $\rho_{\mathscr{M}}$ maps the constant symbol $\mathtt{C} \in \Sigma^* B^\infty 0$ to a stream $\lambda_0(\mathtt{C}) \in B^\infty\Sigma^* 0$ that begins with

$$\tau_0 \xrightarrow{\$} \tau_1 \xrightarrow{a_1} \tau_2 \xrightarrow{a_3} \tau_3 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} \tau_n,$$

where

- each $a_i \in A$ is the last letter of w_{i+1}, i.e., the letter added to the queue in the i-th step of $\mathscr{M}$,
- $\tau_0 = \mathtt{C}$, and $\tau_1,\ldots,\tau_n \in \Sigma^* 0$ are such that each $\tau_i = \mathtt{q}_i(\tau_j)$, where $0 \le j < i$ is such that $i - j = |w_i|$.

(Note that from these properties it follows that $a_j a_{j+1} \cdots a_{i-1} = w_i$.)

Proof. We proceed by induction on n. For the base case $n = 1$, if λ extends $\rho_{\mathscr{M}}$ then, thanks to rule **C**, the stream $\lambda_0(\mathtt{C})$ must begin with:

$$\lambda_0(\mathtt{C}) = \mathtt{C} \xrightarrow{\$} \mathtt{q}_1(\mathtt{C})$$

which satisfies the inductive statement.

For the inductive step, assume that $\mathscr{M}$ makes n steps:

$$q_1, w_1 \longrightarrow\!\!\triangleright\ q_2, w_2 \longrightarrow\!\!\triangleright\ q_3, w_3 \longrightarrow\!\!\triangleright\ \cdots \longrightarrow\!\!\triangleright\ q_n, w_n \longrightarrow\!\!\triangleright\ q_{n+1}, w_{n+1}$$

By the inductive assumption, for any λ that extends $\rho_{\mathscr{M}}$, the stream $\lambda_0(\mathtt{C})$ must begin with:

$$\tau_0 \xrightarrow{\$} \tau_1 \xrightarrow{a_1} \tau_2 \xrightarrow{a_3} \tau_3 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} \tau_n, \tag{7}$$

where $\tau_n = \mathtt{q}_n(\tau_j)$ such that $n - j = |w_n|$, and $w_n = a_j a_{j+1} a_{j+2} \cdots a_{n-1}$.

There are three cases to consider, depending on how the configuration (q_{n+1}, w_{n+1}) is derived from (q_n, w_n):

- $\delta_0(q_n) = (q_{n+1}, a_n)$ and $w_{n+1} = w_n a_n$, for some $a_n \in A$. Then $\rho_{\mathscr{M}}$ includes a corresponding rule **R0**, and if λ extends $\rho_{\mathscr{M}}$ then the initial part (7) in $\lambda_0(\mathtt{C})$ is necessarily extended with $\tau_n \xrightarrow{a_n} \tau_{n+1} = \mathtt{q}_{n+1}(\tau_j)$, and the inductive statement is preserved.
- $\delta_0(q_n)$ is undefined, and $\delta_1(q_n, a_j) = (q_{n+1}, a_n)$ and $w_{n+1} = a_{j+1} a_{j+2} \cdots a_{n-1} a_n$, for some $a_n \in A$. Then $\rho_{\mathscr{M}}$ includes a corresponding rule **R1**, and if λ extends $\rho_{\mathscr{M}}$ then the initial part (7) in $\lambda_0(\mathtt{C})$ is necessarily extended with $\tau_n \xrightarrow{a_n} \tau_{n+1} = \mathtt{q}_{n+1}(\tau_{j+1})$, and the inductive statement is preserved.
- $\delta_0(q_n)$ and $\delta_1(q_n, a_j)$ are undefined, and $\delta_2(q_n, a_j, a_{j+1}) = (q_{n+1}, a_n)$ and $w_{n+1} = a_{j+2} \cdots a_{n-1} a_n$, for some $a_n \in A$. (Note that, since M does not terminate in (q_n, w_n), we know that $n - j \geq 2$.) Then $\rho_{\mathscr{M}}$ includes a corresponding rule **R2**, and if λ extends $\rho_{\mathscr{M}}$ then the initial part (7) in $\lambda_0(\mathtt{C})$ is necessarily extended with $\tau_n \xrightarrow{a_n} \tau_{n+1} = \mathtt{q}_{n+1}(\tau_{j+2})$, and the inductive statement is preserved.

□

Lemma 21 For any QM $\mathscr{M}$ that does not terminate from the initial configuration, the transformation $\rho_{\mathscr{M}}$ is extended by at most one distributive law.

Proof. Consider distributive laws $\lambda, \lambda' : \Sigma^* B^\infty \Longrightarrow B^\infty \Sigma^*$ that both extend $\rho_{\mathscr{M}}$. For any set X, we wish to prove that the component functions $\lambda_X, \lambda'_X : \Sigma^* B^\infty X \to B^\infty \Sigma^* X$ are equal. We prove this by structural induction on terms $t \in \Sigma^* B^\infty X$.

For the first base case, if $t = \sigma \in B^\infty X$ then $\lambda_X(t) = \lambda'_X(t)$ follows immediately from axiom (i) of Definition 9. For the second base case, if $t = \mathtt{C}$ then the equality follows from Lemma 20, since $\mathscr{M}$ makes arbitrarily many steps from the initial configuration.

For the inductive step, we need to prove that for all terms $t \in \Sigma^* B^\infty X$ and states $q \in Q$, if $\lambda_X(t) = \lambda'_X(t)$ then $\lambda_X(\mathtt{q}(t)) = \lambda'_X(\mathtt{q}(t))$. Denote

$$\sigma \quad = \quad \lambda_X(t) \quad = \quad \lambda'_X(t) \quad = \quad \tau_0 \xrightarrow{a_0} \tau_1 \xrightarrow{a_1} \tau_2 \xrightarrow{a_2} \cdots \qquad (\tau_i \in \Sigma^* X).$$

We begin by proving that the desired equality holds when postcomposed with $\pi_{\Sigma^* X} : B^\infty \Sigma^* X \to B \Sigma^* X$, i.e., that the streams $\lambda_X(\mathtt{q}(t))$ and $\lambda'_X(\mathtt{q}(t))$ coincide on their first transitions. This is proved by case analysis similar to that used in the proof of Lemma 20. For example, if $\delta_0(q)$ is undefined and $\delta_1(q, a_0) = (q', b)$

for some $q' \in Q$ and $b \in A$, then $\rho_{\mathscr{M}}$ includes a relevant **R1** rule and if λ and λ' both extend $\rho_{\mathscr{M}}$ then $\lambda_X(\mathtt{q}(t))$ and $\lambda'_X(\mathtt{q}(t))$ must both begin with $\mathtt{q}(\tau_0) \xrightarrow{b} \mathtt{q}'(\tau_1)$.

We proved that for any term $t \in \Sigma^* B^\infty X$ the streams $\lambda_X(t)$ and $\lambda'_X(t)$ coincide on the first transitions, i.e., $\pi\Sigma^* \circ \lambda = \pi\Sigma^* \circ \lambda'$. Hence, by Lemma 16(b), $\lambda = \lambda'$. □

Lemma 22 If a QM $\mathscr{M}$ does not terminate from the initial configuration, then there exists a distributive law that extends $\rho_{\mathscr{M}}$.

Proof. Fix a QM $\mathscr{M}$ that does not terminate from the initial configuration $(q_1, \$)$. We shall define a distributive law λ that extends $\rho_{\mathscr{M}}$. For any set X, begin by defining

$$\lambda_X(\mathtt{C}) \quad = \quad \tau_0 \xrightarrow{a_0} \tau_1 \xrightarrow{a_1} \tau_2 \xrightarrow{a_2} \cdots \quad \in \quad B^\infty \Sigma^* X$$

with $\tau_i \in \Sigma^* X$ and $a_i \in A$ such that:

- $\tau_0 = \mathtt{C}$ and $a_0 = \$$,
- for any $i > 0$, $\tau_i = \mathtt{q}_i(\tau_j)$, where the i-th configuration reached by $\mathscr{M}$ is (q_i, w_i) and $j = i - |w_i|$; moreover, a_j is the first letter of w_i.

To define λ_X on other terms in $\Sigma^* B^\infty X$, note that apart from rule **C**, the entire specification $\rho_{\mathscr{M}}$ is a coGSOS specification, therefore, by Theorem 17, there exists a distributive law $\hat{\lambda}$ that extends all rules of $\rho_{\mathscr{M}}$ apart from **C**. For any term $\mathtt{t} \in \Sigma^* B^\infty X$ where $\mathtt{C}$ does not appear, define $\lambda_X(\mathtt{t})$ to be $\hat{\lambda}_X(\mathtt{t})$. If $\mathtt{C}$ appears in $\mathtt{t}$, replace it with the stream $\lambda_X(\mathtt{C})$ and use $\hat{\lambda}_{\Sigma^* X}$ followed by $B^\infty \mu_X$ on the term obtained.

It is easy to see that λ defined in this manner is natural and satisfies axioms (i)-(iii) of Definition 9 (see also Example 10).

The only remaining axiom is (iv), which in principle could fail if the above procedure, on one of the terms τ_i present in $\lambda_X(\mathtt{C})$, returned a stream that differs from the substream of $\lambda_X(\mathtt{C})$ starting at τ_i. This is, however, not the case, as can be proved by induction on i, using case analysis similar to that used in the proof of Lemma 20. □

Lemma 23 If a QM $\mathscr{M}$ terminates from the initial configuration, then there is no distributive law that extends $\rho_{\mathscr{M}}$.

Proof. Assume to the contrary, that $\mathscr{M}$ terminates after n steps in a configuration (q_n, w_n) and there is a distributive law λ that extends $\rho_{\mathscr{M}}$. By Lemma 20, the stream $\lambda_0(\mathtt{C})$ begins with:

$$\mathtt{C} \xrightarrow{a_1} \tau_1 \xrightarrow{a_2} \tau_2 \xrightarrow{a_3} \cdots \tau_{n-1} \xrightarrow{a_n} \tau_n$$

where $\tau_n = \mathtt{q}_n(\tau_{n-|w_n|})$. Note that $\mathscr{M}$ can terminate in (q_n, w_n) only if w_n has length 1, hence $\tau_n = \mathtt{q}_n(\tau_{n-1})$ and $w_n = a_n$; moreover, $\delta_0(q_n)$ and $\delta_1(q_n, a_n)$ must be undefined.

The remaining argument follows the line of Example 14. Suppose that the next step in $\lambda_0(\mathtt{C})$ is $\tau_n \xrightarrow{a_{n+1}} \tau_{n+1}$, for some $a_{n+1} \in A$ and $\tau_{n+1} \in \Sigma^* 0$. Since $\delta_0(q_n)$ and $\delta_1(q_n, a_n)$ are undefined, $\delta_2(q_n, a_n, a_{n+1}) = (q', b)$ must be defined for some $q' \in Q$ and $b \in A$. As a result, $\rho_{\mathscr{M}}$ contains an **R2** rule:

$$\frac{x \xrightarrow{a_n} y \xrightarrow{a_{n+1}} z}{\mathtt{q}_n(x) \xrightarrow{b} \mathtt{q}'(z)}$$

and, since λ extends $\rho_{\mathscr{M}}$, instantiating x to τ_{n-1} we obtain $b = a_{n+1}$ and $\tau_{n+1} = \mathtt{q}'(\tau_{n+1})$, a contradiction. □

Note that all rules in $\rho_{\mathscr{M}}$ are either GSOS or coGSOS rules; we call specifications with this property *mixed-GSOS specifications*. We arrive at a proof of our Claim from the Introduction:

Theorem 24 For the case of stream systems, it is undecidable whether a given mixed-GSOS specification extends to a unique distributive law.

Proof. Combine Lemmas 21-23 with Theorem 19 □

6 Labelled transition systems

We shall now show how to encode Queue Machines into mixed-GSOS specifications for LTSs, to prove that distributive laws admit no format for $BX = \mathscr{P}_\omega(A \times X)$ either. Since the general idea and most technical details are the same as in the case of stream systems (Section 5), we only sketch the differences between the two cases.

To begin with, note that the set of rules (6) from Section 5 can be read as rules in the mixed-GSOS format for $BX = \mathscr{P}_\omega(A \times X)$. However, taking the same rules for a QM $\mathscr{M}$ would give rise to a biGSOS law that always extends to some distributive law (a counterpart of Lemma 23 would fail). Intuitively, unlike in the case of $BX = A \times X$, a distributive law for $BX = \mathscr{P}_\omega(A \times X)$ is allowed to produce an empty set of successors for a term that corresponds to a terminating configuration of $\mathscr{M}$.

Our solution is to extend the specification (6), now understood as a mixed-GSOS specification for the LTS behaviour, with additional rules:

$$\frac{x \xrightarrow{a} y \quad y \not\xrightarrow{}}{\mathtt{q}(x) \xrightarrow{a} \mathtt{q}(x)} \ (\textbf{R2'}) \tag{8}$$

for $q \in Q$ and $a \in A$. These new rules are included whenever $\delta_0(q)$ and $\delta_1(q,a)$ are undefined. We denote the biGSOS law defined by the extended specification by $\rho_{EXT} : \Sigma B^\infty \Longrightarrow B\Sigma^*$, where $BX = \mathscr{P}_\omega(A \times X)$ and $\Sigma X = 1 + Q \times X$.

For any QM $\mathscr{M}$, the biGSOS law ρ_{EXT} uniquely extends to a distributive law if and only if $\mathscr{M}$ does not terminate from the initial configuration. The proof of this follows the line of Section 5, and we shall only explain the main differences here.

The main technical step in Section 5, Lemma 20, holds in a very similar form:

Lemma 25 For every $n > 0$, if a QM $\mathscr{M}$ makes $n-1$ steps from the initial configuration as in Lemma 20, then every distributive law λ that extends ρ_{EXT} maps the constant symbol $\mathtt{C} \in \Sigma^* B^\infty 0$ to a tree $\lambda_0(\mathtt{C}) \in B^\infty \Sigma^* 0$ that begins with a degenerate tree, i.e., a sequence:

$$\tau_0 \xrightarrow{\$} \tau_1 \xrightarrow{a_1} \tau_2 \xrightarrow{a_3} \tau_3 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} \tau_n$$

where a_i and τ_i are as in Lemma 20.

Proof. By induction on n entirely analogous to the proof of Lemma 20. Intuitively, the initial part of $\lambda_0(\mathtt{C})$ is degenerate because the specification ρ_{EXT} is deterministic, i.e., it only infers one transition from $\mathtt{C}$, and infers at most one transition for $\mathtt{q}(x)$ if x can make at most one transition. □

The next two lemmas are proved entirely analogously to Section 5:

Lemma 26 For an QM $\mathscr{M}$ that does not terminate from the initial configuration, the transformation ρ_{EXT} is extended by at most one distributive law.

Lemma 27 If a QM $\mathscr{M}$ does not terminate from the initial configuration, then there exists a distributive law that extends ρ_{EXT}.

In particular, the distributive law defined in Lemma 27 is exactly as in the proof of Lemma 22, with the streams produced in the latter considered as (degenerate) trees.

The only step that requires some care is Lemma 23, which now takes the form:

Lemma 28 If a QM $\mathscr{M}$ terminates from the initial configuration, then there is no distributive law that extends ρ_{EXT}.

Proof. Assume to the contrary, that $\mathscr{M}$ terminates after n steps in a configuration and there is a distributive law λ that extends $\rho_{\mathscr{M}}$. By Lemma 25, the tree $\lambda_0(\mathtt{C})$ begins with a sequence:

$$\mathtt{C} \xrightarrow{a_1} \tau_1 \xrightarrow{a_2} \tau_2 \xrightarrow{a_3} \cdots \tau_{n-1} \xrightarrow{a_n} \tau_n$$

where, as in the proof of Lemma 23, $\tau_n = \mathtt{q}_n(\tau_{n-1})$, and $\delta_0(q_n)$ and $\delta_1(q_n, a_n)$ are undefined.

What successors can τ_n have in the tree $\lambda_0(\mathtt{C})$? Assume first that is has no successors. Since λ extends ρ_{EXT}, by applying a corresponding rule **R2'** instantiated to $\mathtt{q} = \mathtt{q}_n$, $x = \tau_{n-1}$ and $a = a_n$ we infer that $\tau_n = \mathtt{q}(\tau_{n-1})$ indeed does have at least one successor, which is a contradiction.

Now assume that τ_n has some successors. All these successors are terms in Σ^*0. Some of these successors are minimal, i.e., have the smallest depth of nesting of operations $\mathtt{q}_\mathtt{i}$. Pick one of these minimal successors and call it τ'. Since λ extends ρ_{EXT}, the transition $\tau_n \xrightarrow{b} \tau'$ must be derivable from rules in ρ_{EXT}. The only rule that can be used to this end is a corresponding rule **R2**, instantiated to $\mathtt{q} = \mathtt{q}_n$, $x = \tau_{n-1}$, $y = \tau_n$ and $a = a_n$. But this means that τ_n must have a successor z such that $\tau' = \mathtt{q}'(z)$, which contradicts the minimality of τ'. □

Thus we prove our Claim from the Introduction for the case of LTSs:

Theorem 29 For the case of labeled transition systems ($BX = \mathscr{P}_\omega(A \times X)$), it is undecidable whether a given mixed-GSOS specification extends to a unique distributive law.

Proof. Combine Lemmas 26-28 and Theorem 19. □

7 Related work

We have proved, for the case of stream systems and LTSs, that there is no format for distributive laws of monads over comonads that would be complete for mixed-GSOS specification, i.e., that would cover exactly those mixed-GSOS specification that extend to a distributive law. The specifications used in our proofs are actually coGSOS specifications extended with only one GSOS rule that has no premises. Moreover, the coGSOS rules only uses lookahead of depth 2, and the GSOS rule uses a rule conclusion of height 2. As a result, there is no complete format even for such restricted specifications.

On the other hand, our results do not contradict the existence of formats complete for classes of specifications that do not cover the mixed-GSOS format. Indeed as shown in [13], in the context of LTSs one can combine GSOS and coGSOS but restrict to specifications with positive premises only, and

guarantee the existence of a corresponding distributive law. (Note that specifications used in Section 6 rely on negative rule premises.)

Our proofs can be easily modified to show undecidability of other problems related to operational specifications, some of them phrased without reference to distributive laws. For example, in the case of LTSs, it is undecidable whether a transition system specification (or even a mixed-GSOS specification) has a supported model, a unique supported model, or a unique stable model [6]; the constructions needed for these are minor variations of the one used in Section 6.

In the case of stream systems, our results are related to studies of the productivity of stream definitions [3]. Specifications used in Section 5 can be seen as definitions in the "pure stream specification format" of [3]. Indeed, that format is closely related to stream coGSOS extended with premise-less GSOS rules for constants. In [3] it was proved that productivity of pure stream specifications is decidable for specifications that are data-oblivious, i.e., natural with respect to transition labels. Our specifications are not data-oblivious in that sense. It is easy to use the constructions of Section 5 to prove that productivity of pure stream specifications becomes undecidable without data-obliviousness.

Acknowledgment. We are grateful to Jurriaan Rot for several helpful discussions, and to anonymous referees for spotting embarrassing mistakes both in the content and the presentation of our results.

References

[1] F. Bartels (2004): *On Generalised Coinduction and Probabilistic Specification Formats*. PhD dissertation, CWI, Amsterdam.

[2] B. Bloom, S. Istrail & A. Meyer (1995): *Bisimulation can't be traced*. *Journal of the ACM* 42, pp. 232–268, doi:10.1145/200836.200876.

[3] J. Endrullis, C. Grabmayer, D. Hendriks, A. Isihara & J. Klop (2007): *Productivity of Stream Definitions*. *Fundamentals of Computation Theory*, pp. 274–287, doi:10.1007/978-3-540-74240-1_24.

[4] W. Fokkink (1994): *The Tyft/Tyxt Format Reduces to Tree Rules*. In: *Procs. TACS*, *Lecture Notes in Computer Science* 789, Springer, pp. 440–453, doi:10.1007/3-540-57887-0_109.

[5] W. Fokkink & R. J. van Glabbeek (1996): *Ntyft/ntyxt rules reduce to ntree rules*. *Information and Computation* 126, pp. 1–10, doi:10.1006/inco.1996.0030.

[6] R. J. van Glabbeek (2004): *The meaning of negative premises in transition system specifications II*. *J. Log. Algebr. Program.* 60-61, pp. 229–258, doi:10.1016/j.jlap.2004.03.007.

[7] M. Kick (2002): *Rule Formats for Timed Processes*. In: *Proc. CMCIM'02*, *ENTCS* 68, Elsevier, pp. 12–31, doi:10.1016/S1571-0661(04)80498-5.

[8] B. Klin (2011): *Bialgebras for structural operational semantics: An introduction*. *Theoretical Computer Science* 412(38), pp. 5043–5069, doi:10.1016/j.tcs.2011.03.023. CMCS Tenth Anniversary Meeting.

[9] D. Kozen (1997): *Automata and computability*. Springer, doi:10.1007/978-1-4612-1844-9.

[10] M. Lenisa, J. Power & H. Watanabe (2004): *Category theory for operational semantics*. *Theoretical Computer Science* 327(1-2), pp. 135–154, doi:10.1016/j.tcs.2004.07.024.

[11] S. Mac Lane (1998): *Categories for the Working Mathematician*, second edition. Springer.

[12] J. J. M. M. Rutten (2000): *Universal coalgebra: a theory of systems*. *Theoretical Computer Science* 249, pp. 3–80, doi:10.1016/S0304-3975(00)00056-6.

[13] S. Staton (2008): *General Structural Operational Semantics through Categorical Logic*. In: *Proc. LICS'08*, IEEE Computer Society Press, pp. 166–177, doi:10.1109/LICS.2008.43.

[14] D. Turi & G. D. Plotkin (1997): *Towards a Mathematical Operational Semantics*. In: *Proc. LICS'97*, IEEE Computer Society Press, pp. 280–291, doi:10.1109/LICS.1997.614955.