Science of Computer Programming 172 (2019) 1–26

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Bisimilarity of open terms in stream GSOS ☆

Filippo Bonchi [b,*], Tom van Bussel [a], Matias David Lee [c], Jurriaan Rot [a,*]

[a] ICIS – Radboud University, Nijmegen, Netherlands
[b] Department of Computer Science, University of Pisa, Italy
[c] Université de Lyon, CNRS, ENS de Lyon, UCBL, LIP, 46 Allée d'Italie, Lyon, France

ARTICLE INFO

Article history:
Received 14 February 2018
Received in revised form 28 August 2018
Accepted 22 October 2018
Available online 12 November 2018

Keywords:
Bisimilarity
Streams
Open terms
GSOS
Operational semantics

ABSTRACT

Stream GSOS is a specification format for operations and calculi on infinite sequences. The notion of bisimilarity provides a canonical proof technique for equivalence of closed terms in such specifications. In this paper, we focus on *open terms*, which may contain variables, and which are equivalent whenever they denote the same stream for every possible instantiation of the variables. Our main contribution is to capture equivalence of open terms as bisimilarity on certain Mealy machines, providing a concrete proof technique. Moreover, we introduce an enhancement of this technique, called bisimulation up-to substitutions, and show how to combine it with other up-to techniques to obtain a powerful method for proving equivalence of open terms.

1. Introduction

Structural operational semantics (SOS) can be considered the de facto standard to define programming languages and process calculi. The SOS framework relies on defining a *specification* consisting of a set of operation symbols, a set of labels or actions and a set of inference rules. The inference rules describe the behaviour of each operation, typically depending on the behaviour of the parameters. The semantics is then defined in terms of a labelled transition system over *(closed) terms* constructed from the operation symbols. *Bisimilarity* of closed terms ($\sim$) provides a canonical notion of behavioural equivalence.

It is also interesting to study equivalence of *open terms*, for instance to express properties of program constructors, like the commutativity of a non-deterministic choice operator. The latter can be formalised as the equation $\mathcal{X} + \mathcal{Y} = \mathcal{Y} + \mathcal{X}$, where the left and right hand sides are terms with variables $\mathcal{X}, \mathcal{Y}$. Equivalence of open terms ($\sim_o$) is usually based on $\sim$: for all open terms t_1, t_2

$$t_1 \sim_o t_2 \text{ iff for all closed substitutions } \phi,\ \ \phi(t_1) \sim \phi(t_2). \tag{1}$$

The main problem of such a definition is the quantification over all substitutions: one would like to have an alternative characterisation, possibly amenable to the coinduction proof principle. This issue has been investigated in several works, like [1–7].

In this paper, we continue this line of research, focusing on the simpler setting of *streams*, which are infinite sequences over a fixed data type. More precisely, we consider stream languages specified in the *stream GSOS format* [8], a syntactic rule

☆ The research leading to these results has received funding from the European Research Council (FP7/2007–2013, grant agreement nr. 320571); as well as from the LABEX MILYON (ANR-10-LABX-0070, ANR-11-IDEX-0007), the project PACE (ANR-12IS02001) and the project REPAS (ANR-16-CE25-0011).

* Corresponding authors.
E-mail addresses: filippo.bonchi@unipi.it (F. Bonchi), jrot@cs.ru.nl (J. Rot).

https://doi.org/10.1016/j.scico.2018.10.007
0167-6423/

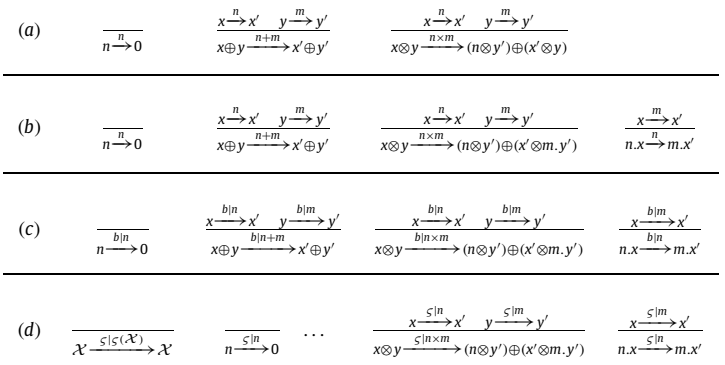


Fig. 1. A stream GSOS specification (a) is transformed first into a monadic specification (b), then in a Mealy specification (c) and finally in a specification for open terms (d). In these rules, n and m range over real numbers, b over an arbitrary set B, $\mathcal{X}$ over variables and ς over substitutions of variables into reals.

format enforcing several interesting properties. We show how to transform a stream specification into a *Mealy machine* specification that defines the operational semantics of open terms. Moreover, a notion of bisimulation – arising in a canonical way from the theory of coalgebras [9] – exactly characterises $\sim_o$ as defined in (1).

Our approach can be illustrated by taking as running example the fragment of the stream calculus [10] presented in Fig. 1(a). The first step is to transform a stream GSOS specification (Section 2) into a *monadic* one (Section 3). In this variant of GSOS specifications, no variable in the source of the conclusion appears in the target of the conclusion. For example, in the stream specification in Fig. 1(a), the rule associated to $\otimes$ is not monadic. The corresponding monadic specification is illustrated in Fig. 1(b). Notice this process requires the inclusion of a family of prefix operators (on the right of Fig. 1(b)) that satisfy the imposed restriction.

The second step – based on [11] – is to compute the *pointwise extension* of the obtained specification (Section 4). Intuitively, we transform a specification of streams with outputs in a set A into a specification of Mealy machines with inputs in an arbitrary set B and outputs in A, by replacing each transition $\xrightarrow{a}$ (for $a \in A$) with a transition $\xrightarrow{b|a}$ for each input $b \in B$. See Fig. 1(c).

In the last step (Section 5), we fix $B = \mathcal{V} \to A$, the set of functions assigning outputs values in A to variables in $\mathcal{V}$. To get the semantics of open terms, it only remains to specify the behaviours of variables in $\mathcal{V}$. This is done with the leftmost rule in Fig. 1(d).

As a result of this process, we obtain a notion of bisimilarity over open terms, which coincides with behavioural equivalence of all closed instances, and provides a concrete proof technique for equivalence of open terms. By relating open terms rather than all its possible instances, this novel technique often enables to use *finite* relations, while standard bisimulation techniques usually require relations of infinite size on closed terms. In Section 6 we further enhance this novel proof technique by studying *bisimulation up-to* [12]. We combine several standard up-to techniques with the notion of *bisimulation up-to substitutions*. These up-to techniques are useful to obtain small relations, and thereby simplify the bisimulation proof technique. For instance, as we show in Section 6, a rather intricate combination of up-to techniques allows us to prove distributivity of shuffle product over sum in stream calculus, that is, $x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$.

Throughout the paper, we exhibit our approach through several examples, such as basic identities in the stream calculus. More generally, since we take stream GSOS as the starting point, our approach is applicable precisely to all *causal* functions on streams: those functions where the n-th element of the output stream only depends on the first n elements of input. This follows from the known result that functions on streams are causal iff they are definable in stream GSOS [13]. For instance, the operations of the stream calculus of Rutten [10] are causal. A typical example of a non-causal function is one which drops every second element of the input stream. Such functions fall outside the scope of the current approach.

An earlier version of this work was presented at FSEN 2017. The current paper extends the proceedings version [14] with proofs (none of which could be included in the proceedings version), and a new section (Section 7) which establishes a connection with very recent work on the *companion* of a functor [15,16]. The companion provides a promising foundation for a general presentation of open terms in *abstract GSOS* [17,8] (a general format for operations on coalgebras formulated in terms of distributive laws, of which stream GSOS is a special case). The aim of the last section is to establish a precise link between the current approach (for streams) and the abstract theory of distributive laws in [15,16], thus placing the current work in a wider context. In particular, we find that the semantics of open terms GSOS introduced in this paper coincides with the semantics that arises canonically from the companion. Further, we show that the Mealy machine over open terms, which arises from the main construction in this paper, can equivalently be obtained through a general correspondence between distributive laws and coalgebras (over a functor category). By its very nature, this section is more technical than the previous sections, and assumes a deeper familiarity with category theory; it is aimed at the reader with an interest in the more general theory of coalgebra and distributive laws.

2. Preliminaries

We define the two basic models that form the focus of this paper: stream systems, that generate infinite sequences (streams), and Mealy machines, that generate output streams given input streams.

Definition 2.1. A *stream system* with outputs in a set A is a pair $(X, \langle o, d\rangle)$ where X is a set of states and $\langle o, d\rangle \colon X \to A \times X$ is a function, which maps a state $x \in X$ to both an *output value* $o(x) \in A$ and to a *next state* $d(x) \in X$. We write $x \xrightarrow{a} y$ whenever $o(x) = a$ and $d(x) = y$.

Definition 2.2. A *Mealy machine* with inputs in a set B and outputs in a set A is a pair (X, m) where X is a set of states and $m \colon X \to (A \times X)^B$ is a function assigning to each $x \in X$ a map $m(x) = \langle o_x, d_x\rangle \colon B \to A \times X$. For all inputs $b \in B$, $o_x(b) \in A$ represents an output and $d_x(b) \in X$ a next state. We write $x \xrightarrow{b|a} y$ whenever $o_x(b) = a$ and $d_x(b) = y$.

We recall the notion of *bisimulation* for both models.

Definition 2.3. Let $(X, \langle o, d\rangle)$ be a stream system. A relation $\mathcal{R} \subseteq X \times X$ is a *bisimulation* if for all $(x, y) \in \mathcal{R}$, $o(x) = o(y)$ and $(d(x), d(y)) \in \mathcal{R}$.

Definition 2.4. Let (X, m) be a Mealy machine. A relation $\mathcal{R} \subseteq X \times X$ is a *bisimulation* if for all $(x, y) \in \mathcal{R}$ and $b \in B$, $o_x(b) = o_y(b)$ and $(d_x(b), d_y(b)) \in \mathcal{R}$.

For both kind of systems, we say that x and y are *bisimilar*, notation $x \sim y$, if there is a bisimulation $\mathcal{R}$ s.t. $x \,\mathcal{R}\, y$.

Stream systems and Mealy machines, as well as the associated notions of bisimulation, are standard examples in the theory of *coalgebras* [9], a mathematical framework that allows to study state-based systems and their semantics at a high level of generality. In the current paper, the theory of coalgebras underlies and enables our main results.

Throughout this paper, we denote by *Set* the category of sets and functions, by $Id \colon Set \to Set$ the identity functor and by $id \colon X \to X$ the identity function on an object X.

Definition 2.5. Given a functor $F \colon Set \to Set$, an *F-coalgebra* is a pair (X, d), where X is a set (called the carrier) and $d \colon X \to FX$ is a function (called the structure). An *F-coalgebra morphism* from $d \colon X \to FX$ to $d' \colon Y \to FY$ is a map $h \colon X \to Y$ such that $Fh \circ d = d' \circ h$.

Stream systems and Mealy machines are F-coalgebras for the functors $FX = A \times X$ and $FX = (A \times X)^B$, respectively.

The semantics of systems modelled as coalgebras for a functor F is provided by the notion of *final coalgebra*. A coalgebra $\zeta \colon Z \to FZ$ is called *final* if for every F-coalgebra $d \colon X \to FX$ there is a unique coalgebra morphism $[\![-]\!] \colon (X, d) \to (Z, \zeta)$. We call $[\![-]\!]$ the *coinductive extension* of d.

Intuitively, the set Z of a final F-coalgebra $\zeta \colon Z \to FZ$ represents the universe of all possible F-behaviours and, for a coalgebra (X, d), $[\![-]\!] \colon X \to Z$ represents its semantics: a function assigning a behaviour to all states in X. This motivates the following definition: given two states $x, y \in X$, x and y are said to be *behaviourally equivalent* iff $[\![x]\!] = [\![y]\!]$. If F preserves weak pullbacks then behavioural equivalence coincides with bisimilarity, i.e., $x \sim y$ iff $[\![x]\!] = [\![y]\!]$ (see [9]). This condition is satisfied by (the functors for) stream systems and Mealy machines. In the sequel, by $\sim$ we hence refer both to bisimilarity and behavioural equivalence.

Final coalgebras for stream systems and Mealy machines will be pivotal for our exposition. We briefly recall them, following [9] and [18]. The set A^ω of streams over A carries a final coalgebra for the functor $FX = A \times X$. For every stream system $\langle o, d\rangle \colon X \to A \times X$, the coinductive extension $[\![-]\!] \colon X \to A^\omega$ assigns to a state $x \in X$ the stream $a_0 a_1 a_2 \dots$ whenever $x \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} \dots$

Recalling a final coalgebra for Mealy machines requires some more care. Given a stream $\beta \in B^\omega$, we write $\beta\restriction_n$ for the prefix of β of length n. A function $\mathfrak{c} \colon B^\omega \to A^\omega$ is *causal* if for all $n \in \mathbb{N}$ and all $\beta, \beta' \in B^\omega$: $\beta\restriction_n = \beta'\restriction_n$ entails $\mathfrak{c}(\beta)\restriction_n = \mathfrak{c}(\beta')\restriction_n$. Intuitively, $\mathfrak{c}$ is causal if the first n symbols of the output stream depends only on the first n symbols of the input stream. The set $\Gamma(B^\omega, A^\omega) = \{\mathfrak{c} \colon B^\omega \to A^\omega \mid \mathfrak{c} \text{ is causal}\}$ carries a final coalgebra for the functor $FX = (A \times X)^B$. For every Mealy machine $m \colon X \to (A \times X)^B$, the coinductive extension $[\![-]\!] \colon X \to \Gamma(B^\omega, A^\omega)$ assigns to each state $x \in X$ and each input stream $b_0 b_1 b_2 \dots \in B^\omega$ the output stream $a_0 a_1 a_2 \dots \in A^\omega$ whenever $x \xrightarrow{b_0|a_0} x_1 \xrightarrow{b_1|a_1} x_2 \xrightarrow{b_2|a_2} \dots$ The interested reader is referred to [18] for more details.

2.1. System specifications

Different kinds of transition systems, like stream systems or Mealy machines, can be specified by means of algebraic specification languages. The syntax is given by an *algebraic signature* Σ, namely a collection of operation symbols $\{f_i \mid i \in I\}$

$$\frac{}{\mathsf{a} \xrightarrow{a} \mathsf{a}}\ \forall a \in A \qquad \frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} y'}{\mathsf{alt}(x, y) \xrightarrow{a} \mathsf{alt}(y', x')}\ \forall a, b \in A$$

Fig. 2. The GSOS-rules of our running example.

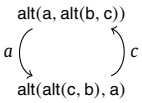


Fig. 3. A stream system.

where each operator f_i has a (finite) arity $n_i \in \mathbb{N}$. For a set X, $T_\Sigma X$ denotes the set of Σ-terms with variables over X. The set of closed Σ-terms is denoted by $T_\Sigma \emptyset$. We omit the subscript when Σ is clear from the context.

A standard way to define the operational semantics of these languages is by means of structural operational semantics (SOS) [19]. In this approach, the semantics of each of the operators is described by syntactic rules, and the behaviour of a composite system is given in terms of the behaviour of its components. We recall *stream GSOS* [8,13], a specification format for stream systems.

Definition 2.6. A *stream GSOS rule* r for a signature Σ and a set A is a rule

$$\frac{x_1 \xrightarrow{a_1} x'_1 \quad \cdots \quad x_n \xrightarrow{a_n} x'_n}{f(x_1, \ldots, x_n) \xrightarrow{a} t} \tag{2}$$

where $f \in \Sigma$ with arity n, $x_1, \ldots, x_n, x'_1, \ldots, x'_n$ are pairwise distinct variables, t is a term built over variables $\{x_1, \ldots, x_n, x'_1, \ldots, x'_n\}$ and $a, a_1, \ldots, a_n \in A$. We say that r is triggered by $(a_1, \ldots, a_n) \in A^n$.

A *stream GSOS specification* is a tuple (Σ, A, R) where Σ is a signature, A is a set of actions and R is a set of stream GSOS rules for Σ and A s.t. for each $f \in \Sigma$ of arity n and each tuple $(a_1, \ldots, a_n) \in A^n$, there is only one rule $r \in R$ for f that is triggered by $(a_1, \ldots, a_n)$.

A stream GSOS specification allows us to extend any given stream system $\langle o, d \rangle \colon X \to A \times X$ to a stream system $\overline{\langle o, d \rangle} \colon TX \to A \times TX$, by induction: the base case is given by $\langle o, d \rangle$, and the inductive cases by the specification. This construction can be defined formally in terms of proof trees, or by coalgebraic means; we adopt the latter approach, which is recalled later in this section.

There are two important uses of the above construction: (A) applying it to the (unique) stream system carried by the empty set $\emptyset$ yields a stream system over closed terms, i.e., of the form $T\emptyset \to A \times T\emptyset$; (B) applying the construction to the final coalgebra yields a stream system of the form $TA^\omega \to A \times TA^\omega$. The coinductive extension $[\![-]\!] \colon TA^\omega \to A^\omega$ of this stream system is, intuitively, the interpretation of the operations in Σ on streams in A^ω.

Example 2.7. Let (Σ, A, R) be a stream GSOS specification where the signature Σ consists of constants $\{\mathsf{a} \mid a \in A\}$ and a binary operation alt. The set R contains the rules in Fig. 2. For an instance of (A), the term $\mathsf{alt}(\mathsf{a}, \mathsf{alt}(\mathsf{b}, \mathsf{c})) \in T\emptyset$ defines the stream system depicted in Fig. 3. For an instance of (B), the operation $\mathsf{alt} \colon A^\omega \times A^\omega \to A^\omega$ maps streams $a_0a_1a_2 \ldots$, $b_0b_1b_2 \ldots$ to $a_0b_1a_2b_2 \ldots$.

Example 2.8. We now consider the specification $(\Sigma, \mathbb{R}, R)$ which is the fragment of the *stream calculus* [20,10] consisting of the constants $n \in \mathbb{R}$ and the binary operators *sum* $\oplus$ and *(convolution) product* $\otimes$. The set R is defined in Fig. 1(a). For an example of (A), consider $n \oplus m \xrightarrow{n+m} 0 \oplus 0 \xrightarrow{0} 0 \oplus 0 \xrightarrow{0} \ldots$. For (B), the induced operation $\oplus \colon \mathbb{R}^\omega \times \mathbb{R}^\omega \to \mathbb{R}^\omega$ is the pointwise sum of streams, i.e., it maps any two streams $n_0n_1 \ldots$, $m_0m_1 \ldots$ to $(n_0 + m_0)(n_1 + m_1) \ldots$.

Definition 2.9. We say that a stream GSOS rule r as in (2) is *monadic* if t is a term built over variables $\{x'_1, \ldots, x'_n\}$. A stream GSOS specification is *monadic* if all its rules are monadic.

The specification of Example 2.7 satisfies the monadic stream GSOS format, while the one of Example 2.8 does not since, in the rules for $\otimes$, the variable y occurs in the arriving state of the conclusion.

The notions introduced above for stream GSOS, as well as the analogous ones for standard (labelled transition systems) GSOS [21], can be reformulated in an abstract framework – the so-called *abstract GSOS* [17,8] – that will be pivotal for the proof of our main result. In this setting, signatures are represented by (polynomial) functors, as follows. A signature Σ presented as a collection $\{f_i \mid i \in I\}$ of operations (with arities $n_i \in \mathbb{N}$) corresponds to the following polynomial functor which, abusing notation, we also denote by Σ:

$$\Sigma X = \coprod_{i\in I} X^{n_i} \qquad \Sigma(g)(f_i(x_1,\ldots,x_{n_i})) = f_i(g(x_1),\ldots,g(x_{n_i}))$$

for every set X and every map $g\colon X \to Y$ where, in the definition of $\Sigma(g)$, we use $f_i(-)$ to refer to the i-th coproduct injection. For instance, the signature Σ in Example 2.7 corresponds to the functor $\Sigma X = A + (X \times X)$, while the signature of Example 2.8 corresponds to the functor $\Sigma X = \mathbb{R} + (X \times X) + (X \times X)$. Models of a signature are seen as algebras for the corresponding functor.

Definition 2.10. Given a functor $F\colon Set \to Set$, an *F-algebra* is a pair (X, d), where X is the *carrier set* and $d\colon FX \to X$ is a function. An *algebra homomorphism* from an F-algebra (X, d) to an F-algebra (Y, d') is a map $h\colon X \to Y$ such that $h \circ d = d' \circ Fh$.

Particularly interesting are *initial algebras*: an F-algebra is called initial if there exists a unique algebra homomorphism from it to every F-algebra. For a functor corresponding to a signature Σ, the initial algebra is $(T\emptyset, \kappa)$ where $\kappa\colon \Sigma T\emptyset \to T\emptyset$ maps, for each $i \in I$, the tuple of closed terms $t_1, \ldots t_{n_i}$ to the closed term $f_i(t_1, \ldots t_{n_i})$. For every set X, we can define in a similar way $\kappa_X\colon \Sigma TX \to TX$. The *free monad over* Σ consists of the endofunctor $T\colon Set \to Set$, mapping every set X to TX, together with the natural transformations $\eta\colon Id \Longrightarrow T$—interpretation of variables as terms—and $\mu\colon TT \Longrightarrow T$—glueing terms built of terms. (We use the standard convention of denoting natural transformations with a double arrow $\Longrightarrow$). Given an algebra $\sigma\colon \Sigma Y \to Y$, for any function $f\colon X \to Y$ there is a unique algebra homomorphism $f^\dagger\colon TX \to Y$ from (TX, κ_X) to (Y, σ).

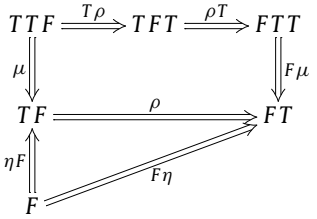
Definition 2.11. An *abstract GSOS specification* (of Σ over F) is a natural transformation $\lambda\colon \Sigma(Id \times F) \Longrightarrow FT$. A *monadic abstract GSOS specification* (in short, monadic specification) is a natural transformation $\lambda\colon \Sigma F \Longrightarrow FT$.

By instantiating the functor F in the above definition to the functor for streams ($FX = A \times X$) one obtains all and only the stream GSOS specifications. Instead, by taking the functor for Mealy machines ($FX = (A \times X)^B$) one obtains the Mealy GSOS format [8]: for the sake of brevity, we do not report the concrete definition here but this notion will be important in Section 5 where, to deal with open terms, we transform stream specifications into Mealy GSOS specifications.

Example 2.12. For every set X, the rules in Example 2.7 define a function $\lambda_X : A + (A \times X) \times (A \times X) \to (A \times T_\Sigma X)$ as follows: each $a \in A$ is mapped to (a, a) and each pair $(a, x'), (b, y') \in (A \times X) \times (A \times X)$ is mapped to $(a, \mathsf{alt}(y', x'))$ [8].

We focus on monadic distributive laws for most of the paper, and since they are slightly simpler than abstract GSOS specifications, we only recall the relevant concepts for monadic distributive laws. However, we note that the concepts below can be extended to abstract GSOS specifications; see, e.g., [22,8] for details.

A monadic abstract GSOS specification induces a distributive law $\rho\colon TF \Longrightarrow FT$, which is a distributive law of the (free) monad (T, η, μ) over the functor F, i.e., it makes the following diagram commute:

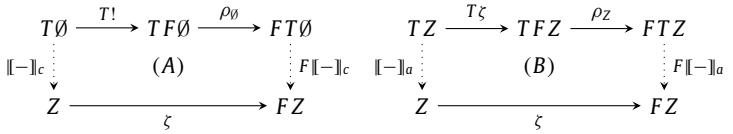


(3)

The distributive law ρ allows us to extend any F-coalgebra $d\colon X \to FX$ to an F-coalgebra on terms:

$$TX \xrightarrow{Td} TFX \xrightarrow{\rho_X} FTX$$

This construction generalises and formalises the aforementioned extension of stream systems to terms by means of a stream GSOS specification. In particular, (A) the unique coalgebra on the empty set $!\colon \emptyset \to F\emptyset$ yields an F-coalgebra on closed terms $T\emptyset \to FT\emptyset$. If F has a final coalgebra (Z, ζ), the unique (hence depicted by a dotted line) morphism $[\![-]\!]_c\colon T\emptyset \to Z$ defines the *semantics of closed terms*.



Further (B), the final coalgebra (Z, ζ) yields a coalgebra on TZ. By finality, we then obtain a T-algebra over the final F-coalgebra, which we denote by $[\![-]\!]_a : TZ \to Z$ and we call it the *abstract semantics*. We define the *algebra induced by* λ as the Σ-algebra $\sigma : \Sigma Z \to Z$ given by

$$\Sigma Z \xrightarrow{\Sigma\eta_Z} \Sigma T Z \xrightarrow{\kappa_Z} T Z \xrightarrow{[\![-]\!]_a} Z\,. \tag{4}$$

3. Making arbitrary stream GSOS specifications monadic

The results presented in the next section are restricted to monadic specifications, but one can prove them for arbitrary GSOS specifications by exploiting some auxiliary operators, introduced in [11] with the name of *buffer*. Theorem 6.2 in Section 6 only holds for monadic GSOS specifications. This does not restrict the applicability of our approach: as we show below, arbitrary stream GSOS specifications can be turned into monadic ones.

Let (Σ, A, R) be a stream GSOS specification. The extended signature $\tilde{\Sigma}$ is given by $\{\tilde{f} \mid f \in \Sigma\} \cup \{a._- \mid a \in A\}$. The set of rules $\tilde{R}$ is defined as follows:

- For all $a, b \in A$, $\tilde{R}$ contains the following rule

$$\frac{x \xrightarrow{b} x'}{a.x \xrightarrow{a} b.x'} \tag{5}$$

- For each rule $r = \dfrac{x_1 \xrightarrow{a_1} x'_1 \quad \cdots \quad x_n \xrightarrow{a_n} x'_n}{f(x_1,\ldots,x_n) \xrightarrow{a} t(x_1,\ldots,x_n,x'_1,\ldots,x'_n)} \in R$, the set $\tilde{R}$ contains

$$\tilde{r} = \frac{x_1 \xrightarrow{a_1} x'_1 \quad \cdots \quad x_n \xrightarrow{a_n} x'_n}{\tilde{f}(x_1,\ldots,x_n) \xrightarrow{a} \tilde{t}(a_1.x'_1,\ldots,a'_n.x'_n,x'_1,\ldots,x'_n)} \tag{6}$$

  where $\tilde{t}$ is the term obtained from t by replacing each $g \in \Sigma$ by $\tilde{g} \in \tilde{\Sigma}$.

The specification $(\tilde{\Sigma}, A, \tilde{R})$ is now monadic and preserves the original semantics as stated by the following result. For a detailed proof, see Appendix A.

Theorem 3.1. *Let (Σ, A, R) be a stream GSOS specification and $(\tilde{\Sigma}, A, \tilde{R})$ be the corresponding monadic one. Then, for all $t \in T_\Sigma\emptyset$, $t \sim \tilde{t}$.*

Example 3.2. Consider the non-monadic specification in Example 2.8. The corresponding monadic specification consists of the rules in Fig. 1 (b) where, to keep the notation light, we used operation symbols f rather than $\tilde{f}$.

In [16], it is conjectured that a variant of Theorem 3.1 goes through at the more general level of abstract/monadic GSOS for arbitrary endofunctors (rather than only streams systems), but a proof of such a general statement is still missing. In [15], it is shown more abstractly that, for polynomial functors on *Set* (such as our stream functor), any operation expressible by a GSOS specification is also expressible by a plain distributive law (of one functor over the other), but that proof is less constructive: it does not result in a concrete new specification, but only shows existence.

4. Pointwise extensions of monadic GSOS specifications

The first step to deal with the semantics of open terms induced by a stream GSOS specification is to transform the latter into a Mealy GSOS specification. We follow the approach in [11] which is defined for arbitrary GSOS but, as motivated in Section 3, we restrict our attention to monadic specifications.

Let (Σ, A, R) be a monadic stream GSOS specification and B some input alphabet. The corresponding monadic Mealy GSOS specification is a tuple $(\Sigma, A, B, \overline{R})$, where $\overline{R}$ is the least set of Mealy rules which contains, for each stream GSOS rule $r = \dfrac{x_1 \xrightarrow{a_1} x'_1 \quad \cdots \quad x_n \xrightarrow{a_n} x'_n}{f(x_1,\ldots,x_n) \xrightarrow{a} t(x'_1,\ldots,x'_n)} \in R$ and $b \in B$, the Mealy rule $\overline{r}_b$ defined by

$$\overline{r}_b = \frac{x_1 \xrightarrow{b|a_1} x'_1 \quad \cdots \quad x_n \xrightarrow{b|a_n} x'_n}{f(x_1,\ldots,x_n) \xrightarrow{b|a} t(x'_1,\ldots,x'_n)}. \tag{7}$$

An example of this construction is shown in Fig. 1 (c).

Recall from Section 2 that any abstract GSOS specification induces a Σ-algebra on the final F-coalgebra. Let $\sigma : \Sigma A^\omega \to A^\omega$ be the algebra induced by the stream specification and $\overline{\sigma} : \Sigma\Gamma(B^\omega, A^\omega) \to \Gamma(B^\omega, A^\omega)$ the one induced by the corresponding Mealy specification. Theorem 4.3, later in this section, informs us that $\overline{\sigma}$ is the *pointwise extension* of σ.

Definition 4.1. Let $g\colon (A^\omega)^n \to A^\omega$ and $\bar{g}\colon (\Gamma(B^\omega, A^\omega))^n \to \Gamma(B^\omega, A^\omega)$ be two functions. We say that $\bar{g}$ is the *pointwise extension* of g iff for all $\mathfrak{c}_1, \ldots, \mathfrak{c}_n \in \Gamma(B^\omega, A^\omega)$ and $\beta \in B^\omega$, $\bar{g}(\mathfrak{c}_1, \ldots, \mathfrak{c}_n)(\beta) = g(\mathfrak{c}_1(\beta), \ldots, \mathfrak{c}_n(\beta))$. This notion is lifted in the obvious way to Σ-algebras for an arbitrary signature Σ.

Example 4.2. Recall the operation $\oplus\colon A^\omega \times A^\omega \to A^\omega$ from Example 2.8 that arises from the specification in Fig. 1 (a) (it is easy to see that the same operation also arises from the monadic specification in Fig. 1(b)). Its pointwise extension $\bar{\oplus}\colon \Gamma(B^\omega, \mathbb{R}^\omega) \times \Gamma(B^\omega, \mathbb{R}^\omega) \to \Gamma(B^\omega, \mathbb{R}^\omega)$ is defined for all $\mathfrak{c}_1, \mathfrak{c}_2 \in \Gamma(B^\omega, \mathbb{R}^\omega)$ and $\beta \in B^\omega$ as $(\mathfrak{c}_1 \bar{\oplus} \mathfrak{c}_2)(\beta) = \mathfrak{c}_1(\beta) \oplus \mathfrak{c}_2(\beta)$. Theorem 4.3 tells us that $\bar{\oplus}$ arises from the corresponding Mealy GSOS specification (Fig. 1(c)).

In [11], the construction in (7) is generalised from stream specifications to arbitrary abstract GSOS. The key categorical tool is the notion of *costrength* for an endofunctor $F\colon Set \to Set$. Given two sets B and X, we first define $\epsilon^b\colon X^B \to X$ as $\epsilon^b(f) = f(b)$ for all $b \in B$. Then, $cs^F_{B,X}\colon F(X^B) \to (FX)^B$ is a natural map in B and X, given by $cs^F_{B,X}(t)(b) = (F\epsilon^b)(t)$. See B for some additional basic properties of costrength.

Now, given a monadic specification $\lambda\colon \Sigma F \Longrightarrow FT$, we define $\bar{\lambda}\colon \Sigma(F^B) \Longrightarrow (FT)^B$ as the natural transformation that is defined for all sets X by

$$\Sigma(FX)^B \xrightarrow{cs^\Sigma_{B,FX}} (\Sigma FX)^B \xrightarrow{\lambda^B_X} (FTX)^B\,. \tag{8}$$

Observe that $\bar{\lambda}$ is also a monadic specification, but for the functor F^B rather than the functor F. The reader can easily check that for F being the stream functor $FX = A \times X$, the resulting $\bar{\lambda}$ is indeed the Mealy specification corresponding to λ as defined in (7).

It is worth to note that the construction of $\bar{\lambda}$ for an arbitrary abstract GSOS $\lambda\colon \Sigma(Id \times F) \Longrightarrow FT$, rather than a monadic one, would not work as in (8). The solution devised in [11] consists of introducing some auxiliary operators as already discussed in Section 3. The following result has been proved in [11] for arbitrary abstract GSOS, with these auxiliary operators. Our formulation is restricted to monadic specifications.

Theorem 4.3. *Let F be a functor with a final coalgebra (Z, ζ), and let $(\bar{Z}, \bar{\zeta})$ be a final F^B-coalgebra. Let $\lambda\colon \Sigma F \Longrightarrow FT$ be a monadic distributive law, and $\sigma\colon \Sigma Z \to Z$ the algebra induced by it. The algebra $\bar{\sigma}\colon \Sigma\bar{Z} \to \bar{Z}$ induced by $\bar{\lambda}$ is a pointwise extension of σ.*

In the theorem above, the notion of pointwise extension should be understood as a generalisation of Definition 4.1 to arbitrary final F and F^B-coalgebras. This generalised notion, that has been introduced in [11], will not play a role for our paper where F is fixed to be the stream functor $FX = A \times X$. Nevertheless, for the sake of completeness, we report its definition in Appendix C, and prove Theorem 4.3 at this general level (C.1).

Example 4.4. We show how to obtain, in the abstract setting, the corresponding Mealy GSOS specification from a stream GSOS specification. To simplify the example we only consider the operator of $\oplus$ of our running example. The syntax, i.e. $\oplus$, defines a functor $\Sigma X = X \times X$ and the behaviour functor of stream systems is $F = \mathbb{R} \times X$. The monadic GSOS specification, i.e. the family of rules for $\oplus$ in Fig. 1 (b), induces a natural transformation

$$\lambda : (\mathbb{R} \times -) \times (\mathbb{R} \times -) \Longrightarrow (\mathbb{R} \times T-)$$

whose X-component is given by:

$$\begin{array}{rccc} \lambda_X\colon & (\mathbb{R} \times X) \times (\mathbb{R} \times X) & \Longrightarrow & \mathbb{R} \times TX \\ & \langle\langle a, x_1\rangle, \langle b, x_2\rangle\rangle & \longmapsto & \langle a + b, \langle x_1, x_2\rangle\rangle \end{array} \tag{9}$$

For defining the pointwise extension $\bar{\oplus}$ of $\oplus$, we have to consider the behaviour functor $F^B X = (\mathbb{R} \times X)^B$. Following the construction of $\bar{\lambda}$, an X-component of a $\bar{\lambda}$ is of the type:

$$\bar{\lambda}_X : (\mathbb{R} \times X)^B \times (\mathbb{R} \times X)^B \Longrightarrow (\mathbb{R} \times TX)^B$$

Let $\phi(b) = \langle \phi_0(b), \phi_1(b)\rangle$ for $\phi \in (\mathbb{R} \times X)^B$ and $b \in B$. By instantiation of (8), $\bar{\lambda}_X$ is defined by

$$\begin{array}{ccc} \langle \phi, \psi\rangle & \in & (\mathbb{R} \times X)^B \times (\mathbb{R} \times X)^B \\ \downarrow & & \downarrow cs^\Sigma_{B,\mathbb{R}\times X} \\ \lambda b.\langle \phi_0(b), \phi_1(b), \psi_0(b), \psi_1(b)\rangle & \in & (\mathbb{R} \times X \times \mathbb{R} \times X)^B \\ \downarrow & & \downarrow \lambda^B_X \\ \lambda b.\langle \phi_0(b) + \psi_0(b), \langle \phi_1(b), \psi_1(b)\rangle\rangle & \in & (\mathbb{R} \times TX)^B \end{array}$$

Using the notation $\phi \xrightarrow{b|\phi_0(b)} \phi_1(b)$ to denote $\phi(b) = \langle \phi_0(b), \phi_1(b) \rangle$ and taking into account the definition of $\bar{\lambda}$, we get what we want, i.e. the rules defining the semantics of $\bar{\oplus}$ are the family of the family of Mealy GSOS rules for $\oplus$ in Fig. 1(c). Notice we have reused the operation symbols f rather than $\bar{f}$ to keep the notation light.

5. Mealy machines over open terms

We now consider the problem of defining a semantics for the set of open terms $T\mathcal{V}$ for a fixed set of variables $\mathcal{V}$. Our approach is based on the results in the previous sections: we transform a monadic GSOS specification for streams with outputs in A into a Mealy machine with inputs in $A^{\mathcal{V}}$ and outputs in A, i.e., a coalgebra for the functor $FX = (A \times X)^{A^{\mathcal{V}}}$. The coinductive extension of this Mealy machine provides the open semantics: for each open term $t \in T\mathcal{V}$ and variable assignment $\psi : \mathcal{V} \to A^\omega$, it gives an appropriate output stream in A^ω. This is computed in a stepwise manner: for an input $\varsigma : \mathcal{V} \to A$, representing "one step" of a variable assignment ψ, we obtain one step of the output stream.

We start by defining a Mealy machine $c : \mathcal{V} \to (A \times \mathcal{V})^{A^{\mathcal{V}}}$ on the set of variables $\mathcal{V}$ as on the left below, for all $\mathcal{X} \in \mathcal{V}$ and $\varsigma \in A^{\mathcal{V}}$:

$$c(\mathcal{X})(\varsigma) = (\varsigma(\mathcal{X}), \mathcal{X}) \qquad \mathcal{X} \circlearrowleft \varsigma|\varsigma(\mathcal{X}) \tag{10}$$

Concretely, this machine has variables as states and for each $\varsigma : \mathcal{V} \to A$ a self-loop, as depicted on the right. Now, let $\lambda : \Sigma(A \times -) \Rightarrow A \times T$ be a monadic abstract stream specification and $\bar{\lambda} : \Sigma((A \times -)^{A^{\mathcal{V}}}) \Rightarrow (A \times T(-))^{A^{\mathcal{V}}}$ be the induced Mealy specification, as defined in (8). As mentioned in Section 2, $\bar{\lambda}$ defines a distributive law $\rho : T((A \times -)^{A^{\mathcal{V}}}) \Rightarrow (A \times T(-))^{A^{\mathcal{V}}}$, which allows to extend c (see (10)) to a coalgebra $m_\lambda : T\mathcal{V} \to (A \times T\mathcal{V})^{A^{\mathcal{V}}}$, given by

$$T\mathcal{V} \xrightarrow{Tc} T(A \times \mathcal{V})^{A^{\mathcal{V}}} \xrightarrow{\rho_{\mathcal{V}}} (A \times T\mathcal{V})^{A^{\mathcal{V}}}. \tag{11}$$

This is the Mealy machine of interest. Intuitively, it is constructed by computing the pointwise extension of the original stream GSOS specification (as in the previous section) and then defining the Mealy machine by induction on terms. The base case is given by (10), and the inductive cases by the (pointwise extended) specification. We first give a few examples.

Example 5.1. Consider the stream specification λ of the operation alt, given in Example 2.7. The states of the Mealy machine m_λ are the open terms $T\mathcal{V}$. The transitions of terms are defined by the set of rules

$$\frac{}{\mathsf{a} \xrightarrow{\varsigma|a} \mathsf{a}} \qquad \frac{x \xrightarrow{\varsigma|a} x' \quad y \xrightarrow{\varsigma|b} y'}{\mathsf{alt}(x, y) \xrightarrow{\varsigma|a} \mathsf{alt}(y', x')} \qquad \text{for all } \varsigma : \mathcal{V} \to A \text{ and } a, b \in A$$

together with the transitions for the variables as in (10). For instance, for each $\mathcal{X}, \mathcal{Y}, \mathcal{Z} \in \mathcal{V}$ and all $\varsigma, \varsigma' : \mathcal{V} \to A$, we have the following transitions in m_λ:

$$\varsigma|\varsigma(\mathcal{X}) \circlearrowleft \mathsf{alt}(\mathcal{X}, \mathsf{alt}(\mathcal{Y}, \mathcal{Z})) \qquad \varsigma'|\varsigma'(\mathcal{Z}) \circlearrowleft \mathsf{alt}(\mathsf{alt}(\mathcal{Z}, \mathcal{Y}), \mathcal{X})$$

Example 5.2. For the fragment of the stream calculus introduced in Example 2.8, the Mealy machine over open terms is defined by the rules in Fig. 1(d). Below we draw the Mealy machines of some open terms that will be useful later.

$$\varsigma|\varsigma(\mathcal{X})+\varsigma(\mathcal{Y}) \circlearrowleft \mathcal{X} \oplus \mathcal{Y} \qquad \varsigma|\varsigma(\mathcal{Y})+\varsigma(\mathcal{X}) \circlearrowleft \mathcal{Y} \oplus \mathcal{X}$$

$$\varsigma|(\varsigma(\mathcal{X})+\varsigma(\mathcal{Y}))+\varsigma(\mathcal{Z}) \circlearrowleft (\mathcal{X} \oplus \mathcal{Y}) \oplus \mathcal{Z} \qquad \varsigma|\varsigma(\mathcal{X})+(\varsigma(\mathcal{Y})+\varsigma(\mathcal{Z})) \circlearrowleft \mathcal{X} \oplus (\mathcal{Y} \oplus \mathcal{Z})$$

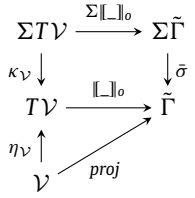
We define the open semantics below by the coinductive extension of m_λ. Let $\tilde{\Gamma} = \Gamma((A^{\mathcal{V}})^\omega, A^\omega)$ be the set of causal functions $\mathfrak{c} : (A^{\mathcal{V}})^\omega \to A^\omega$, which is the carrier of the final coalgebra for the functor $FX = (A \times X)^{A^{\mathcal{V}}}$. Notice that a function $\mathfrak{c} : (A^{\mathcal{V}})^\omega \to A^\omega$ can equivalently be presented as a function $\tilde{\mathfrak{c}} : (A^\omega)^{\mathcal{V}} \to A^\omega$ (swapping the arguments in the domain). Given such a function $\mathfrak{c} : (A^{\mathcal{V}})^\omega \to A^\omega$ and a function $\psi : \mathcal{V} \to A^\omega$, in the sequel, we sometimes abuse of notation by writing $\mathfrak{c}(\psi)$ where we formally mean $\tilde{\mathfrak{c}}(\psi)$.

Definition 5.3. Let $\lambda : \Sigma(A \times -) \Rightarrow A \times T$ be a monadic abstract stream GSOS specification. The *open semantics* of λ is the coinductive extension $[\![-]\!]_o : T\mathcal{V} \to \tilde{\Gamma}$ of the Mealy machine $m_\lambda : T\mathcal{V} \to (A \times T\mathcal{V})^{A^{\mathcal{V}}}$ defined in (11).

Note that the open semantics $[\![-]\!]_o$ assigns to every open term t a causal function on streams. Thus, two terms are behaviourally equivalent (identified by $[\![-]\!]_o$) precisely if they denote the same causal function.

Since $[\![-]\!]_o$ is defined coinductively (as the unique morphism into the final coalgebra), behavioural equivalence of open terms can now be checked by means of bisimulations on Mealy machines (Definition 2.4). We define *open bisimilarity*, denoted by $\sim_o$, as the greatest bisimulation on m_λ. Obviously, for all open terms $t_1, t_2 \in T\mathcal{V}$ it holds that $t_1 \sim_o t_2$ iff $[\![t_1]\!]_o = [\![t_2]\!]_o$. The following result provides another useful characterisation of $[\![-]\!]_o$.

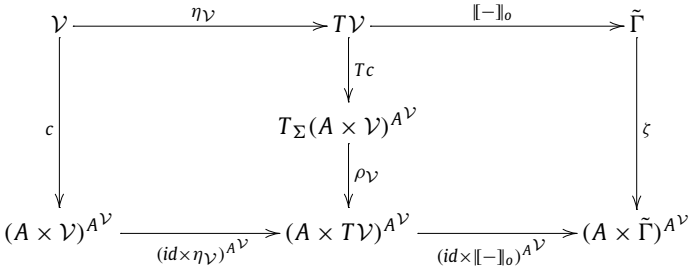
Lemma 5.4. *Let λ be a monadic abstract stream GSOS specification, with induced algebra $\sigma : \Sigma A^\omega \to A^\omega$. Let $\bar{\lambda}$ be the corresponding Mealy specification, with induced algebra $\bar{\sigma} : \Sigma\tilde{\Gamma} \to \tilde{\Gamma}$. Then the open semantics $[\![-]\!]_o$ is the unique homomorphism making the diagram below commute:*



(12)

where η and κ are defined by initiality (Section 2), and for each $\mathcal{X} \in \mathcal{V}$ and $\psi : \mathcal{V} \to A^\omega$, $proj(\mathcal{X})(\psi) = \psi(\mathcal{X})$.

Proof. The open semantics is, by definition, the coinductive extension of m_λ, as depicted on the right below:



where $(\tilde{\Gamma}, \zeta)$ is the final $(A \times -)^{A^{\mathcal{V}}}$-coalgebra. It is a standard fact in the theory of bialgebras that $[\![-]\!]_o$ is an algebra morphism as in the statement of the lemma, so it remains to prove $proj = [\![-]\!]_o \circ \eta$. To this end, observe that $[\![-]\!]_o \circ \eta_{\mathcal{V}}$ is a coalgebra morphism from $c : \mathcal{V} \to (A \times \mathcal{V})^{A^{\mathcal{V}}}$ to the final coalgebra (above diagram; the fact that $\eta_{\mathcal{V}}$ is a coalgebra morphism is again standard). By finality, it suffices to prove that *proj* is a coalgebra morphism. This easily follows from the definition of *proj* and c. □

Observe that, by virtue of Theorem 4.3, the algebra $\bar{\sigma}$ is the pointwise extension of σ. This fact will be useful in the next section to relate $\sim_o$ with bisimilarity on the original stream system.

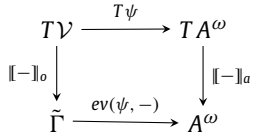
5.1. Abstract, open and closed semantics

Recall from Section 2 the abstract semantics $[\![-]\!]_a : TA^\omega \to A^\omega$ arising as in (B) from a monadic abstract stream specification λ. The following proposition is the key to prove Theorem 5.6 relating open bisimilarity and abstract semantics.

Proposition 5.5. *Let $[\![-]\!]_a$ and $[\![-]\!]_o$ be the abstract and open semantics respectively of a monadic abstract stream GSOS specification λ. For any $t \in T\mathcal{V}$, $\psi : \mathcal{V} \to A^\omega$:*

$$[\![t]\!]_o(\psi) = [\![(T\psi)(t)]\!]_a .$$

Proof. Let $\psi : \mathcal{V} \to A^\omega$. We formulate what we need to prove as commutativity of the following diagram:



where $ev(\psi, -)$ is defined, for $f \in \tilde{\Gamma}$, by

$$ev(\psi, -)(f) = f(\psi)\,. \tag{13}$$

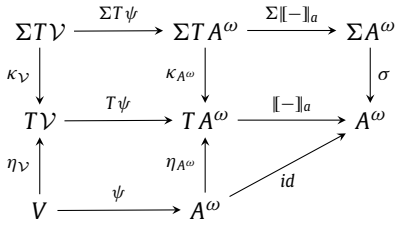
It is convenient below to observe the following equality:

$$ev(\psi, -) = (\tilde{\Gamma} \simeq 1 \times \tilde{\Gamma} \xrightarrow{\psi' \times id} (A^{\mathcal{V}})^{\omega} \times \tilde{\Gamma} \xrightarrow{ev} A^{\omega}) \tag{14}$$

where $\psi' : 1 \to (A^{\mathcal{V}})^{\omega}$ is a transpose of ψ.

Let $\sigma : \Sigma A^{\omega} \to A^{\omega}$ be the algebra induced by the distributive law λ. The proof proceeds by showing that $[\![-]\!]_a \circ T\psi$ and $ev(\psi, -) \circ [\![-]\!]_o$ are both Σ-algebra morphism from the algebra $\kappa_{\mathcal{V}} : \Sigma T\mathcal{V} \to T\mathcal{V}$ to σ, such that $[\![-]\!]_a \circ T\psi \circ \eta_{\mathcal{V}} = ev(\psi, -) \circ [\![-]\!]_o \circ \eta_{\mathcal{V}}$. The equality then follows by uniqueness, see Section 2.1.

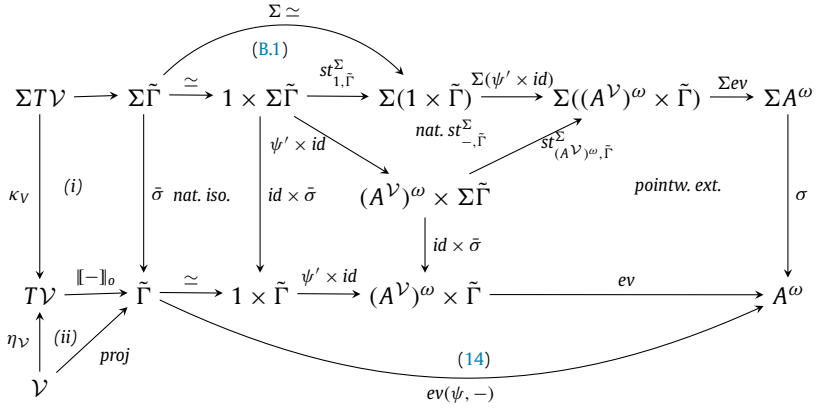
For $[\![-]\!]_a \circ T\psi$, consider the following diagram:



(15)

The squares on the left commute by naturality. The triangle on the right below holds since $[\![-]\!]_a$ is an algebra for the (free) monad T, and the upper square is a standard fact in the theory of bialgebras (it follows from the definition of σ and that $[\![-]\!]_a$ is an algebra for the monad T). This shows that $[\![-]\!]_a \circ T\psi$ is an algebra morphism, and $[\![-]\!]_a \circ T\psi \circ \eta_{\mathcal{V}} = \psi$.

For $ev(\psi, -) \circ [\![-]\!]_o$, consider the diagram below. Here $st^{\Sigma}_{(A^{\mathcal{V}})^{\omega}, \tilde{\Gamma}} : (A^{\mathcal{V}})^{\omega} \times \Sigma\tilde{\Gamma} \to \Sigma((A^{\mathcal{V}})^{\omega} \times \tilde{\Gamma})$ is the *strength* of the functor Σ, see Appendix B for some basic properties which are used in the rest of the proof.



(16)

where $(i), (ii)$ commute by Lemma 5.4 (see also the lemma for the definition of *proj*). Hence $ev(\psi, -) \circ [\![-]\!]_o$ is an algebra morphism, and it only remains to prove that $ev(\psi, -) \circ proj = \psi$:

$$\begin{aligned} ev(\psi, -) \circ proj(\mathcal{X}) &= proj(\mathcal{X})(\psi) && (13) \\ &= \psi(\mathcal{X}) && \text{def. of } proj \end{aligned}$$

From $ev(\psi, -) \circ [\![-]\!]_o \circ \eta_{\mathcal{V}} = ev(\psi, -) \circ proj = \psi = [\![-]\!]_c \circ T\psi \circ \eta_{\mathcal{V}}$ and the fact (shown above) that $ev(\psi, -) \circ [\![-]\!]_o$ and $[\![-]\!]_c \circ T\psi$ are algebra morphisms of the same type, we conclude that they are equal. □

As a simple consequence, we obtain the following characterization of $\sim_o$.

Theorem 5.6. *For all $t_1, t_2 \in T\mathcal{V}$, $[\![t_1]\!]_o = [\![t_2]\!]_o$ iff for all $\psi : \mathcal{V} \to A^{\omega}$: $[\![T\psi(t_1)]\!]_a = [\![T\psi(t_2)]\!]_a$.*

This is one of the main results of this paper: $T\psi(t_1)$ and $T\psi(t_2)$ are expressions in TA^{ω} built from symbols of the signature Σ and streams $\alpha_1, \ldots \alpha_n \in A^{\omega}$. By checking $t_1 \sim_o t_2$ one can prove that the two expressions are equivalent for all possible streams $\alpha_1, \ldots \alpha_n \in A^{\omega}$.

Example 5.7. By using the Mealy machine m_λ in Example 5.1, the relation

$$R = \{(\mathsf{alt}(\mathcal{X}, \mathsf{alt}(\mathcal{Y}, \mathcal{Z})), \mathsf{alt}(\mathcal{X}, \mathsf{alt}(\mathcal{W}, \mathcal{Z}))), (\mathsf{alt}(\mathsf{alt}(\mathcal{Z}, \mathcal{Y}), \mathcal{X}), \mathsf{alt}(\mathsf{alt}(\mathcal{Z}, \mathcal{W}), \mathcal{X}))\}$$

is easily verified to be a bisimulation (Definition 2.4). In particular this shows that $[\![\mathsf{alt}(\mathcal{X}, \mathsf{alt}(\mathcal{Y}, \mathcal{Z}))]\!]_o = [\![\mathsf{alt}(\mathcal{X}, \mathsf{alt}(\mathcal{W}, \mathcal{Z}))]\!]_o$. By Theorem 5.6, we have that $[\![T\psi(\mathsf{alt}(\mathcal{X}, \mathsf{alt}(\mathcal{Y}, \mathcal{Z})))]\!]_a = [\![T\psi(\mathsf{alt}(\mathcal{X}, \mathsf{alt}(\mathcal{W}, \mathcal{Z})))]\!]_a$ for all $\psi : \mathcal{V} \to A^\omega$, i.e.,

$$\mathsf{alt}(\alpha_1, \mathsf{alt}(\alpha_2, \alpha_3)) \sim \mathsf{alt}(\alpha_1, \mathsf{alt}(\alpha_4, \alpha_3)) \text{ for all } \alpha_1, \alpha_2, \alpha_3, \alpha_4 \in A^\omega.$$

The above law can be understood as an equivalence of program schemes stating that one can always replace the stream α_2 by an arbitrary stream α_4, without changing the result.

Example 5.8. By using the Mealy machines in Example 5.2, it is easy to check that both $\{((\mathcal{X} \oplus \mathcal{Y}) \oplus \mathcal{Z}, \mathcal{X} \oplus (\mathcal{Y} \oplus \mathcal{Z}))\}$ and $\{(\mathcal{X} \oplus \mathcal{Y}, \mathcal{Y} \oplus \mathcal{X})\}$ are bisimulations. This means that $[\![(\mathcal{X} \oplus \mathcal{Y}) \oplus \mathcal{Z}]\!]_o = [\![\mathcal{X} \oplus (\mathcal{Y} \oplus \mathcal{Z})]\!]_o$ and $[\![\mathcal{X} \oplus \mathcal{Y}]\!]_o = [\![\mathcal{Y} \oplus \mathcal{X}]\!]_o$. By Theorem 5.6 we obtain associativity and commutativity of $\oplus$:

$$(\alpha_1 \oplus \alpha_2) \oplus \alpha_3 \sim \alpha_1 \oplus (\alpha_2 \oplus \alpha_3) \text{ and } \alpha_1 \oplus \alpha_2 \sim \alpha_2 \oplus \alpha_1 \text{ for all } \alpha_1, \alpha_2, \alpha_3 \in A^\omega.$$

Example 5.9. In a similar way, one can check that $\{((a+b).(\mathcal{X} \oplus \mathcal{Y}), a.\mathcal{X} \oplus b.\mathcal{Y}) \mid a, b \in \mathbb{R}\}$ is a bisimulation. This means that $[\![(a+b).(\mathcal{X} \oplus \mathcal{Y})]\!]_o = [\![a.\mathcal{X} \oplus b.\mathcal{Y}]\!]_o$ for all $a, b \in \mathbb{R}$ and, using again Theorem 5.6, we conclude that $(a+b).(\alpha_1 \oplus \alpha_2) \sim a.\alpha_1 \oplus b.\alpha_2$ for all $\alpha_1, \alpha_2 \in A^\omega$.

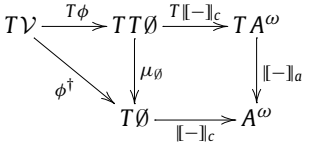
Often, equivalence of open terms is defined by relying on the equivalence of closed terms: two open terms are equivalent iff under all possible closed substitutions, the resulting closed terms are equivalent. For $\sim_o$, this property does not follow immediately by Theorem 5.6, where variables range over streams, i.e., elements of the final coalgebra. One could assume that all the behaviours of the final coalgebra are denoted by some term, however this restriction would rule out most of the languages we are aware of: in particular, the stream calculus that can express only the so-called rational streams [10].

The following theorem, which is the second main result of this paper, only requires that the stream GSOS specification is sufficiently expressive to describe arbitrary finite prefixes. We use that any closed substitution $\phi : \mathcal{V} \to T\emptyset$ defines $\phi^\dagger : T\mathcal{V} \to T\emptyset$ (see Section 2.1).

Theorem 5.10. *Suppose $\lambda : \Sigma(A \times -) \Rightarrow A \times T_\Sigma$ is a monadic abstract stream GSOS specification which contains, for each $a \in A$, the prefix operator $a.-$ as specified in (5) in Section 3. Further, assume $T\emptyset$ is non-empty.*

Let $[\![-]\!]_c$ and $[\![-]\!]_o$ be the closed and open semantics respectively of λ. Then for all $t_1, t_2 \in T\mathcal{V}$: $[\![t_1]\!]_o = [\![t_2]\!]_o$ iff $[\![\phi^\dagger(t_1)]\!]_c = [\![\phi^\dagger(t_2)]\!]_c$ for all $\phi : \mathcal{V} \to T\emptyset$.

Proof. First, we prove that for any $\phi : \mathcal{V} \to T\emptyset$, the following diagram commutes:



(17)

Indeed, commutativity of the left-hand side is a general property of monads, commutativity of the right-hand side a general property of algebras induced by distributive laws.

Now, for the implication from left to right, suppose $[\![t_1]\!]_o = [\![t_2]\!]_o$, and let $\phi : \mathcal{V} \to T\emptyset$. Then

$$[\![\phi^\dagger(t_1)]\!]_c = [\![T([\![-]\!]_c \circ \phi)(t_1)]\!]_a = [\![T([\![-]\!]_c \circ \phi)(t_2)]\!]_a = [\![\phi^\dagger(t_2)]\!]_c .$$

The first and last equality hold by (17). The middle equality holds by Theorem 5.6, instantiated to $\psi = [\![-]\!]_c \circ \phi : \mathcal{V} \to A^\omega$.

For the converse, suppose

$$[\![\phi^\dagger(t_1)]\!]_c = [\![\phi^\dagger(t_2)]\!]_c \tag{18}$$

for all $\phi : \mathcal{V} \to T\emptyset$, and let $\psi : \mathcal{V} \to A^\omega$. We need to prove that $[\![t_1]\!]_o(\psi) = [\![t_2]\!]_o(\psi)$. Let $n \geq 0$, and define $\phi_n : \mathcal{V} \to T\emptyset$ by

$$\phi_n(\mathcal{X}) = \psi(\mathcal{X})(0).\psi(\mathcal{X})(1).(\ldots).\psi(\mathcal{X})(n-1).t$$

for some $t \in T\emptyset$ (assumed to exist); this is indeed a term in $T\emptyset$, since we assumed the presence of the prefix operator. By definition of the prefix operator, it follows that

$$[\![\phi_n(\mathcal{X})]\!]_c \restriction_n = \psi(\mathcal{X}) \restriction_n \tag{19}$$

$$\frac{x \xrightarrow{a} x'}{f(x) \xrightarrow{a} f(x' \oplus x')} \qquad \frac{x \xrightarrow{a} x'}{g(x) \xrightarrow{a} g(x' \oplus x')}$$

Fig. 4. f and g, operators over streams.

$$\frac{x \xrightarrow{\varsigma|a} x'}{f(x) \xrightarrow{\varsigma|a} f(x' \oplus x')} \qquad \frac{x \xrightarrow{\varsigma|a} x'}{g(x) \xrightarrow{\varsigma|a} g(x' \oplus x')}$$

Fig. 5. Pointwise extensions of f and g.

i.e., the first n elements of $[\![\phi_n(\mathcal{X})]\!]_c$ coincide with the first n elements of $\psi(\mathcal{X})$. For all $t \in T\mathcal{V}$, since $[\![t]\!]_o$ is causal, we have

$$[\![t]\!]_o([\![-]\!]_c \circ \phi_n)\restriction_n = [\![t]\!]_o(\psi)\restriction_n . \tag{20}$$

Further, we have, for any $t \in T\mathcal{V}$,

$$[\![t]\!]_o([\![-]\!]_c \circ \phi_n) = [\![T([\![-]\!]_c \circ \phi_n)(t)]\!]_a = [\![\phi_n^\dagger(t)]\!]_c \tag{21}$$

by Proposition 5.5 and (17) respectively. We obtain

$$\begin{aligned}[\![t_1]\!]_o(\psi)\restriction_n = [\![t_1]\!]_o([\![-]\!]_c \circ \phi_n)\restriction_n &= [\![\phi_n^\dagger(t_1)]\!]_c\restriction_n \\ &= [\![\phi_n^\dagger(t_2)]\!]_c\restriction_n = [\![t_2]\!]_o([\![-]\!]_c \circ \phi_n)\restriction_n = [\![t_2]\!]_o(\psi)\restriction_n\end{aligned}$$

by (20), (21) and assumption (18). Since this works for all n, we conclude $[\![t_1]\!]_o(\psi) = [\![t_2]\!]_o(\psi)$ as desired. □

Example 5.11. The specification in Fig. 2 does not include the prefix operator, therefore it does not meet the assumptions of Theorem 5.10. Instead, the monadic GSOS specification in Fig. 1(b) contains the prefix. Recall from Example 5.9 that $(a+b).(\mathcal{X} \oplus \mathcal{Y}) \sim_o a.\mathcal{X} \oplus b.\mathcal{Y}$. Using Theorem 5.10, we can conclude that $(a+b).(t_1 \oplus t_2) \sim a.t_1 \oplus b.t_2$ for all $t_1, t_2 \in T\emptyset$.

6. Bisimulation up-to substitutions

In the previous section, we have shown that bisimulations on Mealy machines can be used to prove equivalences of open terms specified in the stream GSOS format. In this section we introduce *up-to substitutions*, an enhancement of the bisimulation proof method that allows to deal with smaller, often finite, relations. We also show that up-to substitutions can be effectively combined with other well-known up-to techniques such as up-to bisimilarity and up-to context. We note that our results here strongly rely on the specification being monadic: in Appendix D we show that they fail in general for non-monadic specifications.

Intuitively, in a bisimulation up-to substitutions $\mathcal{R}$, the states reached by a pair of states do not need to be related by $\mathcal{R}$, but rather by $\theta(\mathcal{R})$, for some substitution $\theta : \mathcal{V} \to T\mathcal{V}$. We give a concrete example. Suppose we extend the stream calculus of Example 2.8 with the operators f and g defined by the rules in Fig. 4. In Fig. 5, we have the pointwise extensions of these new operators. It should be clear that $f(\mathcal{X}) \sim g(\mathcal{X})$. To try to formally prove $f(\mathcal{X}) \sim g(\mathcal{X})$, consider the relation $\mathcal{R} = \{(f(\mathcal{X}), g(\mathcal{X}))\}$. For all $\varsigma : \mathcal{V} \to A$, there are transitions $f(\mathcal{X}) \xrightarrow{\varsigma|\varsigma(\mathcal{X})} f(\mathcal{X} \oplus \mathcal{X})$ and $g(\mathcal{X}) \xrightarrow{\varsigma|\varsigma(\mathcal{X})} g(\mathcal{X} \oplus \mathcal{X})$. The outputs of both transitions coincide but the reached states are not in $\mathcal{R}$, hence $\mathcal{R}$ is not a bisimulation. However it is a bisimulation up-to substitutions, since the arriving states are related by $\theta(\mathcal{R})$, for some substitution θ mapping $\mathcal{X}$ to $\mathcal{X} \oplus \mathcal{X}$. In fact, without this technique, any bisimulation relating $f(\mathcal{X})$ and $g(\mathcal{X})$ should contain infinitely many pairs.

In order to prove the soundness of this technique, as well as the fact that it can be safely combined with other known up-to techniques, we need to recall some notions of the theory of up-to techniques in lattices from [12]. Given a Mealy machine (X, m), we consider the lattice $(\mathcal{P}(X \times X), \subseteq)$ of relations over X, ordered by inclusion, and the monotone map $\mathbf{b}$: $\mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ defined for all $\mathcal{R} \subseteq X \times X$ as

$$\mathbf{b}(\mathcal{R}) = \{(s,t) \in X \times X \mid \forall b \in B, o_s(b) = o_t(b) \text{ and } d_s(b)\ \mathcal{R}\ d_t(b)\}. \tag{22}$$

It is easy to see that post fixed points of $\mathbf{b}$, i.e., relations $\mathcal{R}$ such that $\mathcal{R} \subseteq \mathbf{b}(\mathcal{R})$, are exactly bisimulations for Mealy machines (Definition 2.4) and that its greatest fixed point is $\sim$.

For a monotone map $\mathbf{f}$: $\mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$, a *bisimulation up-to* $\mathbf{f}$ is a relation $\mathcal{R}$ such that $\mathcal{R} \subseteq \mathbf{bf}(\mathcal{R})$. We say that $\mathbf{f}$ is $\mathbf{b}$-*compatible* if $\mathbf{fb}(\mathcal{R}) \subseteq \mathbf{bf}(\mathcal{R})$ for all relations $\mathcal{R}$. Two results in [12] are pivotal for us: first, if $\mathbf{f}$ is compatible and $\mathcal{R} \subseteq \mathbf{bf}(\mathcal{R})$ then $\mathcal{R} \subseteq \sim$; second if $\mathbf{f}_1$ and $\mathbf{f}_2$ are $\mathbf{b}$-compatible then $\mathbf{f}_1 \circ \mathbf{f}_2$ is $\mathbf{b}$-compatible. The first result informs us that bisimilarity can be proved by means of bisimulations up-to $\mathbf{f}$, whenever $\mathbf{f}$ is compatible. The second result states that compatible up-to techniques can be composed.

We now consider up-to techniques for the Mealy machine over open terms $(T\mathcal{V}, m_\lambda)$ as defined in Section 5. Recall that bisimilarity over this machine is called open bisimilarity, denoted by $\sim_o$. Up-to substitutions is the monotone function $(-)_{\forall\theta} : \mathcal{P}(T\mathcal{V} \times T\mathcal{V}) \to \mathcal{P}(T\mathcal{V} \times T\mathcal{V})$ mapping a relation $\mathcal{R} \subseteq T\mathcal{V} \times T\mathcal{V}$ to

$$(\mathcal{R})_{\forall\theta} = \{(\theta(t_1), \theta(t_2)) \mid \theta : \mathcal{V} \to T\mathcal{V} \text{ and } t_1 \,\mathcal{R}\, t_2\}.$$

Similarly, we define up-to context as the monotone function mapping every relation $\mathcal{R} \subseteq T\mathcal{V} \times T\mathcal{V}$ to its contextual closure $\mathcal{C}(\mathcal{R})$ and up-to (open) bisimilarity as the function mapping $\mathcal{R}$ to $\sim_o \mathcal{R} \sim_o = \{(t_1, t_2) \mid \exists t_1', t_2' \text{ s.t. } t_1 \sim_o t_1' \,\mathcal{R}\, t_2' \sim_o t_2\}$.

Compatibility with **b** of up-to context and up-to bisimilarity hold immediately by the results in [23]. For up-to substitutions, we will next prove compatibility (Theorem 6.2). First, we prove a technical lemma, which states a relation between the derivative of a term after a substitution for a particular input and the derivative of the original term w.r.t. an input constructed based on the substitution and the first input. We use the following notation: given $t \in T\mathcal{V}$, $t(\mathcal{X}_1, \ldots, \mathcal{X}_n)$ denotes that $\mathcal{V}(t) \subseteq \{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$ and $t(t_1, \ldots, t_n)$ the term obtained from t by replacing $\mathcal{X}_i$ by t_i.

Lemma 6.1. *Let $m : T\mathcal{V} \to (A \times T\mathcal{V})^{A^\mathcal{V}}$ be a Mealy machine induced by a monadic Mealy GSOS specification that is the pointwise extension of a monadic stream GSOS specification. Consider a substitution $\theta : \mathcal{V} \to T\mathcal{V}$ and an input $\varsigma : \mathcal{V} \to A$. Then there exists a substitution $\theta' : \mathcal{V} \to T\mathcal{V}$ and input $\varsigma : \mathcal{V} \to A$ such that $o_{\theta(t)}(\varsigma) = o_t(\varsigma')$ and $d_{\theta(t)}(\varsigma) = \theta'(d_t(\varsigma'))$ for all terms $t \in T\mathcal{V}$.*[1]

Proof. We prove this by induction on the structure of the term t, with $\theta' : \mathcal{V} \to T\mathcal{V}$ and $\varsigma' : \mathcal{V} \to A$ given by

(i) $\theta'(\mathcal{X}) = d_{\theta(\mathcal{X})}(\varsigma)$ for all $\mathcal{X} \in \mathcal{V}$, and
(ii) $\varsigma'(\mathcal{X}) = o_{\theta(\mathcal{X})}(\varsigma)$ for all $\mathcal{X} \in \mathcal{V}$.

First we show that the base case holds, i.e. $t = \mathcal{X}$ for some $\mathcal{X} \in \mathcal{V}$. First observe that $o_\mathcal{X}(\varsigma) = \varsigma(\mathcal{X})$ and $d_\mathcal{X}(\varsigma) = \mathcal{X}$ for every input $\varsigma : \mathcal{V} \to A$, as on variables the Mealy machine m is fully defined by the Mealy machine c. From this it follows that $o_{\theta(\mathcal{X})}(\varsigma) = \varsigma'(\mathcal{X}) = o_\mathcal{X}(\varsigma')$, and that $d_{\theta(\mathcal{X})}(\varsigma) = \theta'(\mathcal{X}) = \theta'(d_\mathcal{X}(\varsigma'))$.

Next we prove the inductive case. Consider the case $t = f(t_1, \ldots, t_n)$ where each t_i satisfies the inductive hypothesis. For all $a_1, \ldots, a_n \in A$ and $\varsigma \in A^\mathcal{V}$ there is a rule in the Mealy GSOS specification with the following shape:

$$\frac{\mathcal{X}_1 \xrightarrow{\varsigma|a_1} \mathcal{X}_1' \quad \ldots \quad \mathcal{X}_n \xrightarrow{\varsigma|a_n} \mathcal{X}_n'}{f(\mathcal{X}_1, \ldots, \mathcal{X}_n) \xrightarrow{\varsigma|a^f_{a_1,\ldots,a_n}} t^f_{a_1,\ldots,a_n}(\mathcal{X}_1', \ldots, \mathcal{X}_n')} \tag{23}$$

i.e. the output in the conclusion for f only depends on the outputs of the premises. First we show that $o_{\theta(f(t_1,\ldots,t_n))}(\varsigma) = o_{f(t_1,\ldots,t_n)}(\varsigma')$ using the following calculation:

$$\begin{aligned}
&o_{\theta(f(t_1,\ldots,t_n))}(\varsigma)\\
&= \{\text{by the def. of substitution on terms}\}\\
&o_{f(\theta(t_1),\ldots,\theta(t_n))}(\varsigma)\\
&= \{\text{by the appropriate instantiation of rule (23)}\}\\
&a^f_{o_{\theta(t_1)}(\varsigma),\ldots,o_{\theta(t_n)}(\varsigma)}\\
&= \{\text{by the induction hypothesis}\}\\
&a^f_{o_{t_1}(\varsigma'),\ldots,o_{t_n}(\varsigma')}\\
&= \{\text{by the appropriate instantiation of rule (23)}\}\\
&o_{f(t_1,\ldots,t_n)}(\varsigma')
\end{aligned}$$

Finally, $d_{\theta(f(t_1,\ldots,t_n))}(\varsigma) = \theta'(d_{f(t_1,\ldots,t_n)}(\varsigma'))$ follows from another calculation:

$$\begin{aligned}
&d_{\theta(f(t_1,\ldots,t_n))}(\varsigma)\\
&= \{\text{by def. of substitution on terms}\}\\
&d_{f(\theta(t_1),\ldots,\theta(t_n))}(\varsigma)\\
&= \{\text{by the appropriate instantiation of rule (23)}\}
\end{aligned}$$

[1] Recall $m(t) = \langle o_t, d_t\rangle : A^\mathcal{V} \to A \times T\mathcal{V}$ where $o_t(\varsigma)$ and $d_t(\varsigma)$ are respectively the output and the derivative of $t \in T\mathcal{V}$ for the input $\varsigma \in A^\mathcal{V}$.

$$
\begin{aligned}
&t^f_{o_{\theta(t_1)}(\varsigma),\ldots,o_{\theta(t_n)}(\varsigma)}(d_{\theta(t_1)}(\varsigma),\ldots,d_{\theta(t_n)}(\varsigma))\\
&= \{\text{by induction, } o_{\theta(t_i)}(\varsigma) = o_{t_i}(\varsigma') \text{ for } i = 1,\ldots,n\,\}\\
&t^f_{o_{t_1}(\varsigma'),\ldots,o_{t_n}(\varsigma')}(d_{\theta(t_1)}(\varsigma),\ldots,d_{\theta(t_n)}(\varsigma))\\
&= \{\text{by induction, } d_{\theta(t_i)}(\varsigma) = \theta'(d_{t_i}(\varsigma')) \text{ for } i = 1,\ldots,n\,\}\\
&t^f_{o_{t_1}(\varsigma'),\ldots,o_{t_n}(\varsigma')}(\theta'(d_{t_1}(\varsigma')),\ldots,\theta'(d_{t_n}(\varsigma')))\\
&= \{\text{by def. of substitution on terms}\}\\
&\theta'(t^f_{o_{t_1}(\varsigma'),\ldots,o_{t_n}(\varsigma')}(d_{t_1}(\varsigma'),\ldots,d_{t_n}(\varsigma')))\\
&= \{\text{by the appropriate instantiation of rule (23)}\}\\
&\theta'(d_{f(t_1,\ldots,t_n)}(\varsigma')) \qquad \square
\end{aligned}
$$

Theorem 6.2. *The function* $(-)_{\forall\theta}$ *is* **b***-compatible.*

Proof. For convenience we will write $\mathbf{f} = (-)_{\forall\theta}$. Let $R \subseteq T\mathcal{V} \times T\mathcal{V}$ be a relation, and let $(\theta(s), \theta(t)) \in \mathbf{f}(\mathbf{b}(R))$ for some substitution $\theta : \mathcal{V} \to T\mathcal{V}$ and pair of terms $(s,t) \in \mathbf{b}(R)$. In order to show that $(\theta(s), \theta(t)) \in \mathbf{b}(\mathbf{f}(R))$ we need to prove that

(i) $o_{\theta(s)}(\varsigma) = o_{\theta(t)}(\varsigma)$, and
(ii) $(d_{\theta(s)}(\varsigma), d_{\theta(t)}(\varsigma)) \in \mathbf{f}(R)$,

for every input $\varsigma : \mathcal{V} \to A$.

Now fix ς. By Lemma 6.1 there exists a $\varsigma' : \mathcal{V} \to A$, such that $o_{\theta(t)}(\varsigma) = o_t(\varsigma')$ for all $t \in T\mathcal{V}$. Then, as $(s,t) \in \mathbf{b}(R)$, we have

$$o_{\theta(s)}(\varsigma) = o_s(\varsigma') = o_t(\varsigma') = o_{\theta(t)}(\varsigma).$$

Next we need to show that $(d_{\theta(s)}(\varsigma), d_{\theta(t)}(\varsigma)) \in \mathbf{f}(R)$, i.e. that there exists a substitution $\theta' : \mathcal{V} \to T\mathcal{V}$ and terms $s', t' \in T\mathcal{V}$ such that $\theta'(s') = d_{\theta(s)}(\varsigma)$, $\theta'(t') = d_{\theta(t)}(\varsigma)$, and $(s', t') \in R$. By Lemma 6.1 there exist $\theta' : \mathcal{V} \to T\mathcal{V}$ and $\varsigma' : \mathcal{V} \to A$ such that $d_{\theta(t)}(\varsigma) = \theta'(d_t(\varsigma'))$ for all $t \in T\mathcal{V}$, and thus

$$(d_{\theta(s)}(\varsigma), d_{\theta(t)}(\varsigma)) = (\theta'(d_s(\varsigma')), \theta'(d_t(\varsigma'))).$$

By the assumption that $(s,t) \in \mathbf{b}(R)$, we get that $(d_s(\varsigma'), d_t(\varsigma')) \in R$. So we can conclude that $(d_{\theta(s)}(\varsigma), d_{\theta(t)}(\varsigma)) \in \mathbf{f}(R)$. $\square$

As a consequence of the above theorem and the results in [12], up-to substitutions can be used in combination with up-to bisimilarity and up-to context (as well as any another compatible up-to technique) to prove open bisimilarity. We will show this in the next, concluding example, for which a last remark is useful: the theory in [12] also ensures that if $\mathbf{f}$ is $\mathbf{b}$-compatible, then $\mathbf{f}(\sim) \subseteq \sim$. By Theorem 6.2, this means that $(\sim_o)_{\forall\theta} \subseteq \sim_o$. The same obviously holds for the contextual closure: $\mathcal{C}(\sim_o) \subseteq \sim_o$.

Example 6.3. We prove that the convolution product $\otimes$ distributes over the sum $\oplus$, i.e., $\alpha_1 \otimes (\alpha_2 \oplus \alpha_3) \sim (\alpha_1 \otimes \alpha_2) \oplus (\alpha_1 \otimes \alpha_3)$ for all streams $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{R}^\omega$. By Theorems 5.6 and 6.2, to prove our statement it is enough to show that $\mathcal{R} = \{(\mathcal{X} \otimes (\mathcal{Y} \oplus \mathcal{Z}), (\mathcal{X} \otimes \mathcal{Y}) \oplus (\mathcal{X} \otimes \mathcal{Z}))\}$ is a bisimulation up-to $\sim_o \mathcal{C}(\sim_o (-)_{\forall\theta} \sim_o) \sim_o$.

By rules in Fig. 1(d), for all $\varsigma : \mathcal{V} \to \mathbb{R}$, the transitions of the open terms are

- $\mathcal{X} \otimes (\mathcal{Y} \oplus \mathcal{Z}) \xrightarrow{\varsigma|\varsigma(\mathcal{X})\times(\varsigma(\mathcal{Y})+\varsigma(\mathcal{Z}))} (\varsigma(\mathcal{X}) \otimes (\mathcal{Y} \oplus \mathcal{Z})) \oplus (\mathcal{X} \otimes (\varsigma(\mathcal{Y}) + \varsigma(\mathcal{Z})).(\mathcal{Y} \oplus \mathcal{Z}))$
- $(\mathcal{X} \otimes \mathcal{Y}) \oplus (\mathcal{X} \otimes \mathcal{Z}) \xrightarrow{\varsigma|\varsigma(\mathcal{X})\times\varsigma(\mathcal{Y})+\varsigma(\mathcal{X})\times\varsigma(\mathcal{Z})}$
  $((\varsigma(\mathcal{X}) \otimes \mathcal{Y}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Y}).\mathcal{Y})) \oplus ((\varsigma(\mathcal{X}) \otimes \mathcal{Z}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Z}).\mathcal{Z}))$

For the outputs, it is evident that $\varsigma(\mathcal{X}) \times (\varsigma(\mathcal{Y}) + \varsigma(\mathcal{Z})) = \varsigma(\mathcal{X}) \times \varsigma(\mathcal{Y}) + \varsigma(\mathcal{X}) \times \varsigma(\mathcal{Z})$. For the arriving states we need a few steps, where for all $\varsigma : \mathcal{V} \to \mathbb{R}$ and $\mathcal{X} \in \mathcal{V}$, $\varsigma(\mathcal{X})$ denotes either a real number (used as a prefix) or a constant of the syntax (Example 2.8).

(a) $\mathcal{X} \otimes (\varsigma(\mathcal{Y}).\mathcal{Y} \oplus \varsigma(\mathcal{Z}).\mathcal{Z})\ \mathcal{R}_{\forall\theta}\ (\mathcal{X} \otimes \varsigma(\mathcal{Y}).\mathcal{Y}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Z}).\mathcal{Z})$.
(b) By Example 5.9 and $\mathcal{C}(\sim_o) \subseteq \sim_o$, we have that:
$\mathcal{X} \otimes (\varsigma(\mathcal{Y}) + \varsigma(\mathcal{Z})).(\mathcal{Y} \oplus \mathcal{Z}) \sim_o \mathcal{X} \otimes (\varsigma(\mathcal{Y}).\mathcal{Y} \oplus \varsigma(\mathcal{Z})).\mathcal{Z})$.
(c) By (b) and (a):
$\mathcal{X} \otimes (\varsigma(\mathcal{Y}) + \varsigma(\mathcal{Z})).(\mathcal{Y} \oplus \mathcal{Z}) \sim_o \mathcal{R}_{\forall\theta} \sim_o\ (\mathcal{X} \otimes \varsigma(\mathcal{Y}).\mathcal{Y}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Z}).\mathcal{Z})$.

(d) $\varsigma(\mathcal{X}) \otimes (\mathcal{Y} \oplus \mathcal{Z})\ \mathcal{R}_{\forall\theta}\ (\varsigma(\mathcal{X}) \otimes \mathcal{Y}) \oplus (\varsigma(\mathcal{X}) \otimes \mathcal{Z})$.
(e) Using (d) and (c) with context $\mathcal{C} = \_ \oplus \_$:
$(\varsigma(\mathcal{X}) \otimes (\mathcal{Y} \oplus \mathcal{Z})) \oplus (\mathcal{X} \otimes (\varsigma(\mathcal{Y}) + \varsigma(\mathcal{Z})).(\mathcal{Y} \oplus \mathcal{Z}))$
$\mathcal{C}(\sim_o \mathcal{R}_{\forall\theta} \sim_o)$ $((\varsigma(\mathcal{X}) \otimes \mathcal{Y}) \oplus (\varsigma(\mathcal{X}) \oplus \mathcal{Z})) \oplus ((\mathcal{X} \otimes \varsigma(\mathcal{Y}).\mathcal{Y}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Z}).\mathcal{Z}))$.
(f) By Example 5.8 (associativity and commutativity of $\oplus$) and $(\sim_o)_{\forall\rho} \subseteq \sim_o$:
$((\varsigma(\mathcal{X}) \otimes \mathcal{Y}) \oplus (\varsigma(\mathcal{X}) \oplus \mathcal{Z})) \oplus ((\mathcal{X} \otimes \varsigma(\mathcal{Y}).\mathcal{Y}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Z})).\mathcal{Z}))$
$\sim_o ((\varsigma(\mathcal{X}) \otimes \mathcal{Y}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Y}).\mathcal{Y})) \oplus ((\varsigma(\mathcal{X}) \otimes \mathcal{Z}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Z}).\mathcal{Z}))$.
(g) By (e) and (f):
$(\varsigma(\mathcal{X}) \otimes (\mathcal{Y} \oplus \mathcal{Z})) \oplus (\mathcal{X} \times (\varsigma(\mathcal{Y}) + \varsigma(\mathcal{Z})).(\mathcal{Y} \oplus \mathcal{Z}))$
$\sim_o \mathcal{C}(\sim_o \mathcal{R}_{\forall\theta} \sim_o) \sim_o ((\varsigma(\mathcal{X}) \otimes \mathcal{Y}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Y}).\mathcal{Y})) \oplus ((\varsigma(\mathcal{X}) \otimes \mathcal{Z}) \oplus (\mathcal{X} \otimes \varsigma(\mathcal{Z}).\mathcal{Z}))$.

7. A "familiar" construction of the Mealy machine of open terms

In the previous sections, we transformed a monadic abstract GSOS specification into a 'Mealy machine of open terms', using the pointwise extension. While it is still unclear if—and how—our approach generalises beyond streams and Mealy machines to arbitrary coalgebras, in the current section we make the first steps toward such a generalisation by connecting our work to recent developments in the theory of distributive laws [15,16]. These developments provide a 'semantics of distributive laws', by organising them into a category with a final object, the so-called *companion*.

The main aim of this section is to connect the concrete construction that we provided in the current paper to this abstract work on distributive laws and the companion. On the one hand, we show that the construction of a 'Mealy machine of open terms' and the association of causal functions to open terms (Section 5), is an instance of a more general construction in the theory of distributive laws. On the other hand, it provides a concrete case study for some of the abstract techniques in [15,16]. We therefore view the technical development in the current section as a potentially useful step towards a more general coalgebraic theory of open terms.

The connection with [15,16] can be anticipated in a nutshell. Let $[Set, Set]$ be the category of Set endofunctors and natural transformations between them. In [15,16], a functor $\mathbb{F}\colon [Set, Set] \to [Set, Set]$ is defined with the property that $\mathbb{F}$-coalgebras are in bijective correspondence with distributive laws over a fixed functor $F\colon Set \to Set$ (we use the blackboard font $\mathbb{F}$ to distinguish from the Set endofunctors used throughout the paper, denoted by plain capital letters). When instantiated to the functor for stream systems $FX = A \times X$, a distributive law $\rho\colon TF \Rightarrow FT$, corresponds via the above construction to an $\mathbb{F}$-coalgebra, i.e., a natural transformation $\widehat{\rho}\colon T \Rightarrow \mathbb{F}(T)$. Interestingly, this natural transformation $\widehat{\rho}$, at the component $\mathcal{V}$, turns out to be the Mealy machine m_λ of open terms defined in (11). In this way, the Mealy machine m_λ is thus constructed via the correspondence between distributive laws and coalgebras.

In the remainder of this section we make this more precise. We start with the basic necessary definitions.

Definition 7.1. Given an endofunctor $F\colon Set \to Set$, the category $\mathsf{DL}(F)$ has pairs (Σ, λ) as objects, where $\Sigma\colon Set \to Set$ is a functor and $\lambda\colon \Sigma F \Longrightarrow F\Sigma$ is a distributive law, and a morphism from (Σ_1, λ_1) to (Σ_2, λ_2) is a natural transformation $\theta\colon \Sigma_1 \Longrightarrow \Sigma_2$ such that $\lambda_2 \circ \theta F = F\theta \circ \lambda_1$, see [24–27]. The final object $(C, \gamma\colon CF \Longrightarrow FC)$ in $\mathsf{DL}(F)$ is called the *companion* of F, if it exists. The companion is thus characterised by the property that for every distributive law $\lambda\colon \Sigma F \Longrightarrow F\Sigma$ there exists a unique natural transformation $\kappa\colon \Sigma \Longrightarrow F$ making the diagram below commute:

$$\begin{array}{ccc} \Sigma F & \xRightarrow{\kappa F} & CF \\ {\scriptstyle\lambda}\Big\Downarrow & & \Big\Downarrow{\scriptstyle\gamma} \\ F\Sigma & \xRightarrow[F\kappa]{} & FC \end{array} \tag{24}$$

See [15,16] for a systematic study of the above notion of companion. In the following theorem, $\mathsf{coalg}(\mathbb{F})$ denotes the category of $\mathbb{F}$-coalgebras and coalgebra morphisms between them.

Theorem 7.2 *([16]). There exists a functor $\mathbb{F}\colon [Set, Set] \to [Set, Set]$ which gives a one-to-one correspondence between distributive laws $\lambda\colon \Sigma F \Longrightarrow F\Sigma$ and coalgebras $\widehat{\lambda}\colon \Sigma \Longrightarrow \mathbb{F}(\Sigma)$ (natural in F). In particular, there is an isomorphism of categories:*

$$\mathsf{DL}(F) \cong \mathsf{coalg}(\mathbb{F})\,. \tag{25}$$

In [16], $\mathbb{F}$ is called the *familiar* of F, and is characterised abstractly using right Kan extensions. The theorem below provides a concrete characterization for the familiar of the functor $FX = A \times X$ and for the isomorphism of Theorem 7.2. First, $\mathbb{F}\colon [Set, Set] \to [Set, Set]$ is defined for all $\Sigma\colon Set \to Set$ as

$$(A \times \Sigma -)^{A^-}\colon Set \to Set\,. \tag{26}$$

Second, given a natural transformation of the form $\theta\colon HF \Longrightarrow FG$, we define $\widehat{\theta}\colon H \Longrightarrow \mathbb{F}(G)$, for all sets X, as

$$HX \xrightarrow{Hc_X} H(A \times X)^{A^X} \xrightarrow{cs^H_{A^X, A\times X}} (H(A \times X))^{A^X} \xrightarrow{(\theta_X)^{A^X}} (A \times GX)^{A^X} \tag{27}$$

where the natural transformation $c\colon Id \Longrightarrow (A \times -)^{A^-}$ is given for all sets X and $x \in X$ by

$$c_X(x)(\varsigma) = (\varsigma(x), x). \tag{28}$$

Theorem 7.3. *The familiar of $FX = A \times X$ is $\mathbb{F}\colon [Set, Set] \to [Set, Set]$ defined as in (26). The assignment $\theta \mapsto \widehat{\theta}$ defined in (27), restricted to the case where $H = G$, extends to a functor which witnesses the isomorphism in (25).*

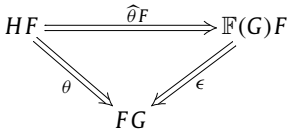
Proof. In [16], it is shown that the familiar of an endofunctor F is given as

$$\mathbb{F}(G) = \mathsf{Ran}_F(FG).$$

Now we show that, given an endofunctor G, the right Kan extension $\mathsf{Ran}_F(FG)$ is given by$(A \times G-)^{A^-}$, with the counit $\epsilon\colon \mathbb{F}(G)F \Longrightarrow FG$ given by

$$\epsilon_X(a\colon A^{A\times X} \to A \times G(A \times X)) = (id \times G\pi_2)(a(\pi_1))$$

on a set X, which is easily seen to be natural. Thus we have to show that for every $H\colon Set \to Set$ and $\theta\colon HF \Longrightarrow FG$, there exists a unique $\widehat{\theta}\colon H \Longrightarrow \mathbb{F}(G)$ making the diagram below commute:



First we show that the diagram above commutes with the $\widehat{\theta}$ defined in (27), i.e. we show that $\epsilon \circ \widehat{\theta}F = \theta$. Given $t \in GX$ we have:

$$\begin{aligned}
&\epsilon_X(\widehat{\theta}_{FX}(t))\\
&= \{\text{by definition of } \epsilon \text{ (counit)}\}\\
&FG(\pi_2)(\widehat{\theta}_{FX}(t)(\pi_1))\\
&= \{\text{by definition of } \widehat{\theta}\}\\
&FG(\pi_2)(\theta^{A^{FX}}_{FX}(cs^H_{A^{FX}, FFX}(H(c_{FX})(t)))(\pi_1))\\
&= \{\text{by definition of the functor } (-)^{A^X}\}\\
&FG(\pi_2)(\theta_{FX}(cs^H_{A^{FX}, FFX}(H(c_{FX})(t))(\pi_1))))\\
&= \{\text{by definition of costrength}\}\\
&FG(\pi_2)(\theta_{FX}(H(\epsilon^{\pi_1})(H(c_{FX})(t))))\\
&= \{\text{by naturality of } \theta\}\\
&\theta_X(H(F\pi_2 \circ \epsilon^{\pi_1} \circ c_{FX})(t))\\
&= \{\text{as } F\pi_2 \circ \epsilon^{\pi_1} \circ c_{FX} = id\}\\
&\theta_X(t)
\end{aligned}$$

Next we show that $\widehat{\theta}$ is the unique natural transformation making the diagram above commute. Let $\kappa : H \Longrightarrow \mathbb{F}(G)$ such that $\epsilon \circ \kappa F = \theta$. Given $t \in HX$ and $\varsigma\colon X \to A$ we have:

$$\begin{aligned}
&\widehat{\theta}_X(t)(\varsigma)\\
&= \{\text{by definition of } \widehat{\theta}\}\\
&\theta_X(cs^H_{A^X, FX}(H(c_X)(t))(\varsigma))\\
&= \{\text{by assumption}\}\\
&\epsilon_X(\kappa_{FX}(cs^H_{A^X, FX}(H(c_X)(t))(\varsigma)))\\
&= \{\text{by definition of } \epsilon \text{ (counit)}\}
\end{aligned}$$

$$
\begin{aligned}
&FG(\pi_2)(\kappa_{FX}(cs^H_{A^X,FX}(H(c_X)(t))(\varsigma))(\pi_1))\\
&= \{\text{by definition of costrength}\}\\
&FG(\pi_2)(\kappa_{FX}(H(\epsilon^\varsigma \circ c_X)(t))(\pi_1))\\
&= \{\text{by naturality of } \kappa\}\\
&FG(\pi_2)(\mathbb{F}(G)(\epsilon^\varsigma \circ c_X)(\kappa_X(t))(\pi_1))\\
&= \{\text{by definition of } \mathbb{F}(G) \text{ on morphisms}\}\\
&FG(\pi_2)(FG(\epsilon^\varsigma \circ c_X)(\kappa_X(t)(\pi_1 \circ \epsilon^\varsigma \circ c_X)))\\
&= \{\text{as } \pi_2 \circ \epsilon^\varsigma \circ c_X = id \text{ and } \pi_1 \circ \epsilon^\varsigma \circ c_X = \varsigma\}\\
&\kappa_X(t)(\varsigma)
\end{aligned}
$$

So $\kappa = \widehat{\theta}$ and thus $\widehat{\theta}$ is the natural transformation from H to $\mathbb{F}(G)$ such that $\epsilon \circ \widehat{\theta}F = \theta$. So $\mathbb{F}(G) = \mathsf{Ran}_F FG \cong (A \times G-)^{A^-}$. □

Now take a monadic stream GSOS specification $\lambda\colon \Sigma F \Rightarrow FT$ and the corresponding distributive law $\rho\colon TF \Rightarrow FT$. Via (27), ρ corresponds to the $\mathbb{F}$-coalgebra $\widehat{\rho}\colon T \Rightarrow (A \times T-)^{A^-}$, which at the component $\mathcal{V}$, is a function of the shape

$$\widehat{\rho}_{\mathcal{V}}\colon T\mathcal{V} \to (A \times T\mathcal{V})^{A^{\mathcal{V}}},$$

namely, a Mealy machine with input $A^{\mathcal{V}}$, output A and state space $T\mathcal{V}$. This is exactly the Mealy machine m_λ for open terms defined in (11), Section 5:

Corollary 7.4. *With ρ and λ as above, we have $\widehat{\rho}_{\mathcal{V}} = m_\lambda$.*

Proof. Recall from Section 5 that m_λ is defined as

$$T\mathcal{V} \xrightarrow{Tc} T(A \times \mathcal{V})^{A^{\mathcal{V}}} \xrightarrow{\bar{\rho}_{\mathcal{V}}} (A \times T\mathcal{V})^{A^{\mathcal{V}}} \tag{29}$$

where $\overline{\rho}$ is the distributive law corresponding to the pointwise extension $\bar{\lambda}$ of λ, and c in the above coincides with $c_{\mathcal{V}}$ in (28). Now, by Lemma A.11, we have

$$\bar{\rho}_{\mathcal{V}} = \left(T(A \times \mathcal{V})^{A^{\mathcal{V}}} \xrightarrow{cs^T_{A^{\mathcal{V}},A\times\mathcal{V}}} (T(A \times \mathcal{V}))^{A^{\mathcal{V}}} \xrightarrow{(\rho_{\mathcal{V}})^{A^{\mathcal{V}}}} (A \times T\mathcal{V})^{A^{\mathcal{V}}} \right). \tag{30}$$

Combining (29) and (30) yields $\widehat{\rho}_{\mathcal{V}}$ by definition of $\widehat{\rho}$. Hence $\widehat{\rho}_{\mathcal{V}} = m_\lambda$. □

A further observation sheds light on the relationship with the companion. In [15], it is shown that companions of polynomial endofunctors on *Set* exist, and a characterisation is given using a generalisation of the notion of a causal function. For the functor $FX = A \times X$, this notion ends up to be the usual definition of causal function, given in Section 2. To characterise the companion, it is convenient—following Section 5—to say $\mathfrak{c}\colon (A^\omega)^X \to A^\omega$ is causal if the associated $\bar{\mathfrak{c}}\colon (A^X)^\omega \to A^\omega$ (swapping arguments) is causal in the usual sense (Section 2). Below, we denote the set of all such causal functions by $\Gamma((A^\omega)^X, A^\omega)$. Then, using the results in [15], the companion $(C, \gamma\colon CF \Longrightarrow FC)$ of the stream system functor $FX = A \times X$ is given by

$$CX = \Gamma((A^\omega)^X, A^\omega)\,.$$

Since (C, γ) is a final object in $\mathsf{DL}(F)$, there is a unique natural transformation $\kappa\colon (T, \rho) \Longrightarrow (C, \gamma)$. Its component at $\mathcal{V}$ is exactly the open semantics from Definition 5.3, as stated formally by the following result.

Theorem 7.5. *Let $\lambda\colon \Sigma F \Rightarrow FT$ be an abstract monadic stream GSOS specification, and let $\rho\colon TF \Rightarrow FT$ be the corresponding distributive law. Let $\kappa\colon T \Rightarrow C$ be the unique morphism from (T, ρ) to (C, γ). Then $\kappa_{\mathcal{V}} = [\![-]\!]_o$.*

We give a self-contained proof of this fact, together with a concrete characterisation of the companion (including the associated distributive law), in Appendix E.

8. Conclusions, related and future work

In this paper we have studied the semantics of open terms specified in the stream GSOS format. Our recipe consists in translating the stream specification into a Mealy specification giving semantics to all open terms. Remarkably, this semantics equates two open terms if and only if they are equivalent under all possible interpretations of variables as streams (Theorem 5.6) or under the interpretation of variables as closed terms (Theorem 5.10). Furthermore, semantic equivalence can be checked by means of the bisimulation proof method enhanced with a technique called up-to substitutions (Theorem 6.2).

Two considerations are now in order. First, the main advantage of using up-to substitutions rather than more standard coinductive techniques (e.g. [13,8]) is that the former may allow finite relations to witness equivalence of open terms (see Example 6.3), while the latter would often require infinite relations containing all their possible instantiations. We expect this difference to be relevant for equivalence checking algorithms; we leave as future work to properly investigate this issue.

Second, the correspondences in Theorems 5.6 and 5.10 are far from being expected. Indeed, for GSOS specifying labelled transition systems, the notions of bisimilarity of open terms proposed in literature (like the formal hypothesis in [1], the hypothesis preserving in [2], the loose and the strict in [3], the rule-matching in [4]) are sound w.r.t. the semantics obtained by the interpretation of variables as closed terms, but *not* complete. Such incompleteness seems to be natural for the authors of [3] since, they say, in modern open systems, software can be partially specified, executed and then instantiated during its execution. From a coalgebraic perspective, the difficulties in having a complete coinductive characterisation for the semantics of open semantics seems to arise from non-determinism: passing from streams to labelled transition systems means from the coalgebraic outlook to move from the functor $FX = A \times X$ to $FX = \mathcal{P}(A \times X)$.

Endrullis et al. [28] consider bisimulation up-to techniques between stream terms, in order to improve coinductive methods in Coq. Their terms include variables, and up-to substitution is also used there. It is shown that up-to substitution is sound, in combination with other techniques such as up-to context (for which causality is assumed; this is equivalent to GSOS-definability [13]). The use of variables suggests that open terms may possibly be treated by the techniques in [28] as well; however, bisimulations between open terms are not explicitly mentioned there. Moreover, the approach is different: in particular, it is not based on the construction via Mealy machines. A more precise understanding of the connection to the current paper is left for future work.

In this sense, our work can be considered as a first concrete step towards a (co)algebraic understanding of the semantics of open terms in the general setting of abstract GSOS [17,8]. Orthogonally to the current paper, there is the more abstract perspective on distributive laws and abstract GSOS in the recent papers [16,15], which potentially is of use for bisimilarity of open terms. In Section 7, we connected these two approaches, showing how the current concrete work on streams fits in the more abstract perspective offered by these papers. This is a promising starting point for a general coalgebraic theory of bisimilarity of open terms, for GSOS specifications for arbitrary functors. We leave the development of such a theory for future work.

One of the potential benefits of such a theory might be a general coalgebraic theory of complete axiomatizations. The work of Silva on Kleene coalgebra [29] is a successful step in this direction, but its scope is limited to regular behaviours. Interestingly enough, one of the first completeness results for regular behaviours [30] already makes use of the semantics of open terms. This is indeed considered in the field to be one of the standard techniques to prove (ω)-completeness for axiomatisations, see the survey in [31].

To conclude, it is worth to stress the fact that our approach is confined to GSOS specifications. Extending it to more expressive formats, such as tyft/tyxt [32] or ntyft/ntyxt [33], seems to be rather challenging. Indeed, while GSOS specification have a neat categorical description in terms of distributive laws [17,8], we are not aware of similar results for more expressive formats. Furthermore, while bisimilarity is guaranteed to be a congruence by these formats, bisimulations up-to context is in general not compatible: a counterexample for the soundness of up-to context and bisimilarity for the tyft format can be found in [12].

Appendix A. Proof of Theorem 3.1

Theorem A.1 *(Th. 3.1). Let* (Σ, A, R) *be a stream GSOS specification and* $(\tilde{\Sigma}, A, \tilde{R})$ *be the corresponding monadic one. Then, for all* $t \in T_\Sigma \emptyset$, $t \sim \tilde{t}$.

In order to prove Theorem 3.1 we have to recall the notion of *disjoint extension*.

Definition A.2. A (stream) GSOS specification $S' = (\Sigma', A, R')$ is a disjoint extension of the (stream) GSOS specification $S = (\Sigma, A, R)$ if $\Sigma \subseteq \Sigma'$, $R \subseteq R'$ and R' adds no new rules for operators in Σ.

Because R' adds no new rules for operators in Σ we have that the behaviour of $t \in T_\Sigma \emptyset$ is the same for both specifications S and its disjoint extension S'.

Let $S = (\Sigma, R, A)$ be a stream GSOS specification and let $\tilde{S} = (\tilde{\Sigma}, \tilde{R}, A)$ be its translation to a monadic GSOS specification, see Section 3. If we consider the stream GSOS specification $S + \tilde{S} = (\Sigma \cup \tilde{\Sigma}, A, R \cup \tilde{R})$ we have that $S + \tilde{S}$ is a disjoint extension of both S and $\tilde{S}$. Let $\tilde{t}$ be the term obtained from $t \in T_\Sigma \emptyset$ by replacing each occurrence of $f \in \Sigma$ in t by $\tilde{f} \in \tilde{\Sigma}$.

Taking into account $S + \tilde{S}$ we will prove that for all $t \in T_\Sigma\emptyset$ we have that $\tilde{t} \in T_{\tilde{\Sigma}}\emptyset$ is such that $t \sim \tilde{t}$, see Lemma A.8. Because $S + \tilde{S}$ is a disjoint extension of $\tilde{S}$, we can ensure that Theorem 3.1 is sound.

In order to prove the result we define a relation $\mathcal{R} \subseteq T_{\Sigma\cup\tilde{\Sigma}}\emptyset \times T_{\Sigma\cup\tilde{\Sigma}}\emptyset$ and we prove that this relation is a bisimulation up-to bisimilarity. Let $\mathcal{R} \subseteq T_{\Sigma\cup\tilde{\Sigma}}\emptyset \times T_{\Sigma\cup\tilde{\Sigma}}\emptyset$ be the smallest relation satisfying:

($\mathcal{R}$-i) $(c, \tilde{c}) \in \mathcal{R}$ for each constant $c \in \Sigma$.
($\mathcal{R}$-ii) $(t, a.\tilde{t'}) \in \mathcal{R}$ if $t \in T_\Sigma\emptyset$ and $t \xrightarrow{a} t'$.
($\mathcal{R}$-iii) $(f(t_1, \ldots, t_n), \tilde{f}(s_1, \ldots, s_n)) \in \mathcal{R}$ whenever $t_i \,\mathcal{R}\, s_i$ for all $i = 1, \ldots, n$ and $f \in \Sigma$.

By structural induction on t and considering the different cases based on the definition of $\mathcal{R}$ we can prove the following results

Lemma A.3. *If $(t, s) \in \mathcal{R}$ then $t \in T_\Sigma\emptyset$ and $s \in T_{\tilde{\Sigma}}\emptyset$.*

Lemma A.4. *For all $t \in T_\Sigma\emptyset$, $(t, \tilde{t}) \in \mathcal{R}$.*

Lemma A.5. *For all $t(x_1, \ldots, x_m) \in T_\Sigma\mathcal{V}$, $t_1, \ldots, t_m \in T_\Sigma\emptyset$, $s_1, \ldots, s_m \in T_{\tilde{\Sigma}}\emptyset$, if $t_i \,\mathcal{R}\, s_i$ for $i = 1, \ldots, m$ then $t(t_1, \ldots, t_m) \,\mathcal{R}\, \tilde{t}(s_1, \ldots, s_m)$.*

In addition we have:

Lemma A.6. *For all $t \in T_{\Sigma\cup\tilde{\Sigma}}\emptyset$, if $t \xrightarrow{a} t'$ then $t \sim a.t'$.*

Proof. Suppose that $t_0, t_1, \ldots \in T_{\Sigma\cup\tilde{\Sigma}}\emptyset$ and $a_0, a_1, \ldots \in A$ are s.t. $t = t_0$, $t' = t_1$, $a_0 = a$ and $t_0 \xrightarrow{a_0} t_1 \xrightarrow{a_1} t_2 \xrightarrow{a_2} \ldots$. Define $\mathcal{S} = \{(t_i, a_i.t_{i+1}) \mid i = 0, 1, \ldots\}$. Relation $\mathcal{S}$ is a bisimulation given that for all i, $t_i \xrightarrow{a_i} t_{i+1}$, $a_i.t_{i+1} \xrightarrow{a_i} a_{i+1}.t_{i+2}$ by rule (5) and $t_{i+1} \,\mathcal{S}\, a_{i+1}.t_{i+2}$. Finally $(t, a.t') = (t_0, a_0.t_1) \in \mathcal{S}$ □

Lemma A.7. *$\mathcal{R}$ is a bisimulation up-to bisimilarity w.r.t. $S + \tilde{S}$.*

Proof. We prove $\mathcal{R}$ is a bisimulation up-to bisimilarity by structural induction on $t \in T_\Sigma\emptyset$ where $(t, s) \in \mathcal{R}$ for some s. Considering $t \in T_\Sigma\emptyset$ is enough because of Lemma A.3.

Suppose that $t = c$ is a constant. By Definition 2.6, $c \xrightarrow{a} t'$ with $t' \in T_\Sigma\emptyset$ iff there is axiom r, i.e. rule without premises, $c \xrightarrow{a} t'$. We have two cases to analyze for s as a consequence of ($\mathcal{R}$-i) and ($\mathcal{R}$-ii): (i) Case $s = \tilde{c}$. By construction of $\tilde{S}$ there is an axiom $\tilde{r}$ s.t. $\tilde{r} = \tilde{c} \xrightarrow{a} \tilde{t'}$. By Lemma A.4, $t' \,\mathcal{R}\, \tilde{t'}$. (ii) Case $s = a.\tilde{t'}$. Suppose that $\tilde{t'} \xrightarrow{b} s''$ then $\tilde{t'} \sim b.s''$ by Lemma A.6. By (5), $a.\tilde{t'} \xrightarrow{a} b.s''$. Finally, $t' \sim t' \,\mathcal{R}\, \tilde{t'} \sim b.s''$ because $\sim$ is reflexive, Lemma A.4 and Lemma A.6.

For the inductive case we consider $t = f(t_1, \ldots, t_n)$. Suppose $f(t_1, \ldots, t_n) \xrightarrow{a} t'$. We have two cases to analyze as a consequence of ($\mathcal{R}$-ii) and ($\mathcal{R}$-iii). Case ($\mathcal{R}$-ii) follows similarly to its counterpart in the base case of the inductive proof. For the case ($\mathcal{R}$-iii) we consider $(f(t_1, \ldots, t_n), \tilde{f}(s_1, \ldots, s_n)) \in \mathcal{R}$ with $t_i \,\mathcal{R}\, s_i$ for all $i = 1, \ldots, n$. Suppose that

$$f(t_1, \ldots, t_n) \xrightarrow{a} t(t_1, \ldots t_n, t'_1, \ldots, t'_n)$$

because of an instantiation of the rule (2), then $t_i \xrightarrow{a_i} t'_i$ for each $i = 1, \ldots, n$. By induction, for each i, $t_i \xrightarrow{a_i} t'_i$, $s_i \xrightarrow{a_i} s'_i$ and there are $\hat{t}_i$ and $\hat{s}_i$ s.t.

$$t'_i \sim \hat{t}_i \qquad \hat{s}_i \sim s'_i \tag{A.1}$$

$$\hat{t}_i \,\mathcal{R}\, \hat{s}_i \tag{A.2}$$

Using rule (6) and for each i, $s_i \xrightarrow{a_i} s'_i$, we can derive the following transition

$$\tilde{f}(s_1, \ldots, s_n) \xrightarrow{a} \tilde{t}(a_1.s'_1, \ldots a_n.s_n, s'_1, \ldots, s'_n)$$

To conclude the proof we have to prove that there are $\hat{t}$ and $\hat{s}$ such that

$$t(t_1, \ldots t_n, t'_1, \ldots, t'_n) \sim \hat{t} \,\mathcal{R}\, \hat{s} \sim \tilde{t}(a_1.s'_1, \ldots a_n.s'_n, s'_1, \ldots, s'_n)$$

To prove this, notice first that for each i

$$t_i \sim a_i.t'_i \sim a_i.\hat{t}_i \qquad a_i.\hat{s}_i \sim a_i.s'_i \tag{A.3}$$

$$a_i.\hat{t}_i \,\mathcal{R}\, a_i.\hat{s}_i \tag{A.4}$$

The left side equation of (A.3) is a consequence of Lemma A.6 and (A.1), taking into account that $\sim$ is a congruence for the prefix operators. The right side equation of (A.3) is also a consequence of this last fact. Equation (A.4) is a consequence of ($\mathcal{R}$-iii) and (A.2).

Define $\hat{t} = t(a_1.\hat{t}_1, \ldots, a_n.\hat{t}_n, \hat{t}_1, \ldots, \hat{t}_n)$ and $\hat{s} = \tilde{t}(a_1.\hat{s}_1, \ldots, a_n.\hat{s}_n, \hat{s}_1, \ldots, \hat{s}_n)$. Recall that $\sim$ is a congruence for all operator defined using the stream GSOS specification format and this property can be extended to arbitrary context constructed using these operators. Taking into account this fact and (A.1) and (A.3) we get $t(t_1, \ldots t_n, t'_1, \ldots, t'_n) \sim \hat{t}$ and $\hat{s} \sim \tilde{t}(a_1.s'_1, \ldots a_n.s_n, s'_1, \ldots, s'_n)$. Finally $\hat{t}\ \mathcal{R}\ \hat{s}$ because of Lemma A.5, (A.2) and (A.4). □

Lemma A.8. *Let S and $\tilde{S}$ be, resp., a stream GSOS specification and its encoding in monadic GSOS specification. For the GSOS specification $S + \tilde{S}$ we have $t \sim \tilde{t}$ for all $t \in T_\Sigma \emptyset$.*

Proof. The result is a straight consequence of Lemmas A.4 and A.7 □

Appendix B. Strength and costrength

The categorical notions of strength and costrength are recalled here, as they play an important role from Section 4 onwards. The *strength* of an endofunctor F on Set is a map $st^F_{A,X} \colon A \times FX \to F(A \times X)$ natural in A and X, defined by $st^F_{A,X}(a, t) = (F\eta^a)(t)$ where $\eta^a : X \to A \times X$ is given by $\eta^a(x) = (a, x)$. The *costrength* of F is a map $cs^F_{A,X} \colon F(X^A) \to (FX)^A$ natural in A and X, defined by $cs^F_{A,X}(t)(a) = (F\epsilon^a)(t)$ where $\epsilon^a : X^A \to X$ is given by $\epsilon^a(f) = f(a)$.

We recall several basic properties that will only be used in proofs, and can be safely skipped by the reader. First of all, the following diagrams commute:

$$\begin{array}{ccc} A \times \Sigma V & \xrightarrow{st^\Sigma_{A,V}} & \Sigma(A \times V) \\ {\scriptstyle \pi_2}\downarrow & \swarrow_{\Sigma\pi_2} & \\ \Sigma V & & \end{array} \qquad \begin{array}{ccc} 1 \times \Sigma V & \xrightarrow{st^\Sigma_{1,V}} & \Sigma(1 \times V) \\ {\scriptstyle \simeq}\downarrow & \swarrow_{\Sigma \simeq} & \\ \Sigma V & & \end{array} \tag{B.1}$$

For the triangle on the left-hand side, we have $\Sigma\pi_2 \circ st^\Sigma_{A,V}(a, t) = \Sigma\pi_2 \circ \Sigma\eta^a(t) = \Sigma(\pi_2 \circ \eta^a)(t) = t = \pi_2(a, t)$. The triangle on the right-hand side is a special case.

We will also use that costrength is natural in the endofunctor involved, i.e., for any natural transformation $\gamma : F \Rightarrow G$ and any sets A, X, the following diagram commutes:

$$\begin{array}{ccc} F(X^A) & \xrightarrow{cs^F_{A,X}} & (FX)^A \\ {\scriptstyle \gamma_{X^A}}\downarrow & & \downarrow{\scriptstyle (\gamma_X)^A} \\ G(X^A) & \xrightarrow{cs^G_{A,X}} & (GX)^A \end{array} \tag{B.2}$$

An analogous property holds for strength, see [11, Appendix A].

Finally, when T is a monad, the costrength $cs^T_{A,-}$ is always a distributive law of the monad T over the functor $(-)^A$ (Diagram (3) with $F = (-)^A$), see, e.g., [34, Example 2].

Appendix C. The pointwise extension

We recall the general definition of pointwise extension. This requires a few preliminary notions, in particular of a map *ev*; the reader can safely skip these general definitions and move immediately to the instance where the functor at hand is the one for streams, for which we give a concrete characterisation.

Definition A.9. Let $s : Y \to B \times Y$ be an $(B \times -)$-coalgebra. For an F^B-coalgebra $m : X \to (FX)^B$, define the F-coalgebra

$$s \ltimes m : Y \times X \to F(Y \times X)$$

to be the function composition:

$$Y \times X \xrightarrow{s \times m} (B \times Y) \times (FX)^B \xrightarrow{\simeq} Y \times (B \times (FX)^B)$$
$$\xrightarrow{id_Y \times \epsilon_{FX}} Y \times FX \xrightarrow{st^F_{Y,X}} F(Y \times X)$$

Let $\langle Z, \zeta \rangle$ be a final F-coalgebra. By finality, for all $s : Y \to B \times Y$ and $m : X \to (FX)^B$, there is a unique F-coalgebra morphism $[\![-]\!]_{s \ltimes m} : Y \times X \to Z$:

$$\begin{array}{ccc} Y \times X & \xrightarrow{[\![-]\!]_{s \ltimes m}} & Z \\ {\scriptstyle s \ltimes m}\downarrow & & \downarrow{\scriptstyle \zeta} \\ F(Y \times X) & \xrightarrow{F[\![-]\!]_{s \ltimes m}} & FZ \end{array} \tag{C.1}$$

We instantiate the above by taking s to be the final $(B \times -)$-coalgebra $(B^\omega, \langle \mathsf{hd}, \mathsf{tl} \rangle)$, and m to be the final F^B-coalgebra $(\bar{Z}, \bar{\zeta})$; ev is then defined to be the coinductive extension below.

$$\begin{array}{ccc} B^\omega \times \bar{Z} & \xrightarrow{ev} & Z \\ {\scriptstyle \langle \mathsf{hd}, \mathsf{tl} \rangle \ltimes \bar{\zeta}}\downarrow & & \downarrow{\scriptstyle \zeta} \\ F(B^\omega \times \bar{Z}) & \xrightarrow{Fev} & FZ \end{array} \tag{C.2}$$

In the current paper, we are interested in the case that $Z = A^\omega$ consists of streams (the final coalgebra of stream systems), and $\bar{Z} = \Gamma(B^\omega, A^\omega)$ consists of causal functions (the final coalgebra of Mealy machines). In that case, $ev\colon B^\omega \times \Gamma(B^\omega, A^\omega) \to A^\omega$ is given concretely by

$$ev(\alpha, f) = f(\alpha) \tag{C.3}$$

as reported in [11, Example 4.3].

Definition A.10. For a signature Σ, an algebra $\bar{\sigma} : \Sigma\bar{Z} \to \bar{Z}$ is a pointwise extension of $\sigma : \Sigma Z \to Z$ if the following diagram commutes:

$$\begin{array}{ccccc} B^\omega \times \Sigma\bar{Z} & \xrightarrow{st^{\Sigma}_{B^\omega, \bar{Z}}} & \Sigma(B^\omega \times \bar{Z}) & \xrightarrow{\Sigma ev} & \Sigma(Z) \\ {\scriptstyle id \times \bar{\sigma}}\downarrow & & & & \downarrow{\scriptstyle \sigma} \\ B^\omega \times \bar{Z} & & \xrightarrow{ev} & & Z \end{array} \tag{C.4}$$

This generalises the concrete instance for streams in Definition 4.1.

C.1. Proof of Theorem 4.3

We need the following lemma to prove Theorem 4.3:

Lemma A.11. *Let $\lambda\colon \Sigma F \Longrightarrow FT$ be a monadic abstract GSOS specification, with pointwise extension $\bar{\lambda}\colon \Sigma F^B \Rightarrow (FT)^B$. Let $\bar{\rho}\colon TF^B \Rightarrow (FT)^B$ be the distributive law induced by $\bar{\lambda}$. Then*

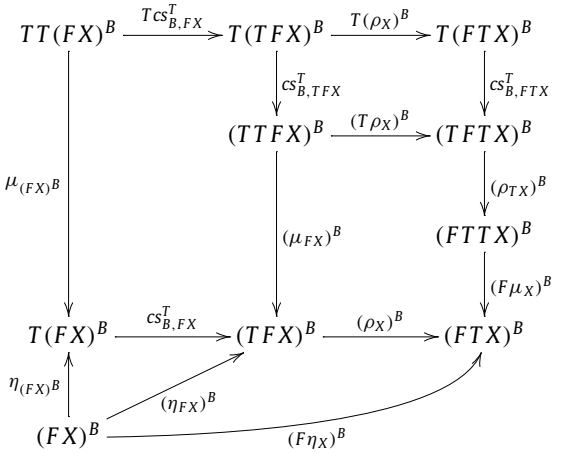
$$\bar{\rho} = \left(T(FX)^B \xrightarrow{cs^T_{B,FX}} (TFX)^B \xrightarrow{(\rho_X)^B} (FTX)^B \right)$$

where $\rho\colon TF \Rightarrow FT$ is the distributive law induced by λ.

Proof. Let $\bar{\rho}' = (T(FX)^B \xrightarrow{cs^T_{B,FX}} (TFX)^B \xrightarrow{(\rho_X)^B} (FTX)^B)$. By [22, Lemma 3.4.27], $\bar{\rho}$ is the unique distributive law satisfying $\bar{\lambda} = \bar{\rho} \circ \kappa_{F^B} \circ \Sigma\eta_{F^B}$ (this is an instance of the one-to-one correspondence between (monadic) GSOS specifications and distributive laws). Hence, it suffices to show the following properties:

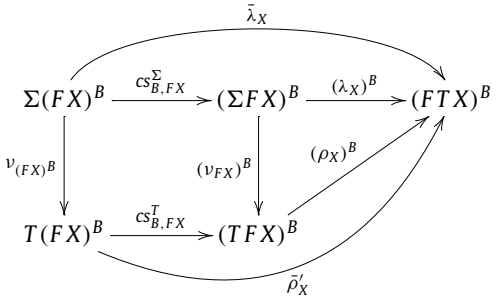
1. $\bar{\rho}'$ is a distributive law of monad over functor;
2. $\bar{\lambda} = \bar{\rho}' \circ \kappa_{F^B} \circ \Sigma\eta_{F^B}$.

For 1., consider the following diagram:



The left part of the diagram commutes since $cs^T_{B,-}$ is a distributive law of the monad T over the functor $(-)^B$, the right part follows since ρ is a distributive law of the monad T over the functor F (and the top-right square by naturality of $cs^T_{B,-}$).

For 2., consider the following diagram:



(C.5)

where $\nu = \kappa \circ \Sigma\eta$. The square commutes by (B.2), the triangle on to the right since ρ is the distributive law extending λ, and the upper and lower shapes by definition of $\bar{\lambda}$ and $\bar{\rho}'$ respectively. □

This allows us to prove Theorem 4.3.

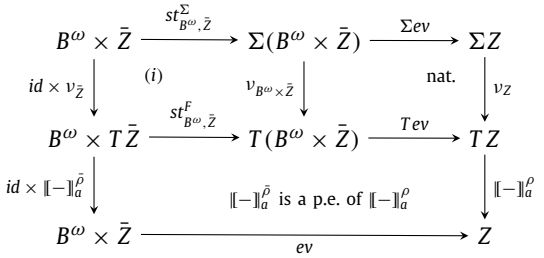
Proof. [Proof of Theorem 4.3] Let $[\![-]\!]^{\bar{\rho}}_a : T\bar{Z} \to \bar{Z}$ and $[\![-]\!]^{\rho}_a : TZ \to Z$ be the algebras arising by finality from $\bar{\rho}$ and ρ respectively, as defined in diagram (B) at the end of Section 2.1.

Let $\nu = \kappa \circ \Sigma\eta$. The algebras $\sigma : \Sigma Z \to Z$ and $\bar{\sigma} : \Sigma\bar{Z} \to \bar{Z}$ are defined respectively by $[\![-]\!]^{\rho}_a \circ \nu_Z$ and $[\![-]\!]^{\bar{\rho}}_a \circ \nu_{\bar{Z}}$. We need to prove that the latter is a pointwise extension of the former.

By Lemma A.11, the distributive law $\bar{\rho}$ induced by $\bar{\lambda}$ is given by

$$\bar{\rho} = (T(FX)^B \xrightarrow{cs^T_{B,X}} (TFX)^B \xrightarrow{(\rho_X)^B} (FTX)^B)\,.$$

Hence, by [11, Theorem 6.1], $[\![-]\!]^{\bar{\rho}}_a$ is a pointwise extension of $[\![-]\!]^{\rho}_a$, i.e., the lower rectangle in the diagram below commutes:



(C.6)

The square (i) is naturality of strength (see Section Appendix B), and the square on the right commutes by naturality of ν. □

Appendix D. Non-compatibility for non-monadic specifications

In our approach, we first transform an arbitrary stream GSOS specification into a monadic one and then we construct the corresponding Mealy specification. One could transform directly an arbitrary specification into a Mealy specification following the approach in [11], but Theorem 6.2 in Section 6 would not go through. In this appendix we shortly explain why.

The pointwise extension of GSOS specification that is not monadic requires the introduction of a family of auxiliary operators, the so-called *buffer operators*. For a stream GSOS specification $\lambda\colon \Sigma(Id \times F) \Longrightarrow FT$ with $F = A \times X$, there is a Mealy GSOS specification $\bar{\lambda}\colon \bar{\Sigma}(Id \times (F)^{A^{\mathcal{V}}}) \Longrightarrow (F\bar{T})^{A^{\mathcal{V}}}$ that pointwise extends λ (see [11, Theorem 6.3]). Notice that the signature used for the Mealy machine is not the original Σ but $\bar{\Sigma}$. In our setting, this signature is the extension of Σ with the buffer operators $\varsigma\triangleright$, for each $\varsigma : \mathcal{V} \to A$. The extended specification $\bar{\lambda}$ defines their semantics as follows:

$$\frac{x \xrightarrow{\varsigma|a} x'}{\varsigma \triangleright x \xrightarrow{\varsigma'|a} \varsigma' \triangleright x'}.$$

In addition, for each rule r with shape (2) of the stream GSOS specification and $\varsigma : \mathcal{V} \to A$, the Mealy machines has a rule r' defined by

$$r' = \frac{x_1 \xrightarrow{\varsigma|a_1} x_1' \quad \cdots \quad x_n \xrightarrow{\varsigma|a_n} x_n'}{f(x_1,\dots,x_n) \xrightarrow{\varsigma|a} t(\varsigma \triangleright x_1,\dots,\varsigma \triangleright x_n, x_1',\dots,x_n')}$$

By following this approach, open bisimilarity is *not* preserved by substitution.

For example, consider stream systems over reals and the family of "bad" prefix operators $a.x$, for each $a \in \mathbb{R}$. The following left and right rules represent respectively, a stream GSOS specification and its corresponding Mealy specification.

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{}{a.x \xrightarrow{\varsigma|a} \varsigma \triangleright x}$$

For a fixed $a \in \mathbb{R}$, let $\hat{\varsigma} : \mathcal{V} \to \mathbb{R}$ be such that $\hat{\varsigma}(\mathcal{X}) = a$ and $\hat{\varsigma}(\mathcal{Y}) \neq 0$. Then $a.\mathcal{X}$ and $\hat{\varsigma} \triangleright \mathcal{X}$ are in $\sim_o$ because

$$a.\mathcal{X} \xrightarrow{\varsigma|a} \varsigma \triangleright \mathcal{X} \text{ and } \hat{\varsigma} \triangleright \mathcal{X} \xrightarrow{\varsigma|\hat{\varsigma}(\mathcal{X})} \varsigma \triangleright \mathcal{X}.$$

Consider now substitution $\theta : \mathcal{V} \to T\mathcal{V}$ such that $\theta(\mathcal{X}) = \mathcal{X} \oplus \mathcal{Y}$, where $\oplus$ is defined as in Fig. 1(a). Then $\theta(a.\mathcal{X})$ and $\theta(\hat{\varsigma} \triangleright \mathcal{X})$ are not bisimilar because $\theta(a.\mathcal{X}) = a.(\mathcal{X} \oplus \mathcal{Y}) \xrightarrow{\varsigma|a} \varsigma \triangleright (\mathcal{X} \oplus \mathcal{Y})$, $\theta(\hat{\varsigma} \triangleright \mathcal{X}) = \hat{\varsigma} \triangleright (\mathcal{X} \oplus \mathcal{Y}) \xrightarrow{\varsigma|\hat{\varsigma}(\mathcal{X})+\hat{\varsigma}(\mathcal{Y})} \hat{\varsigma} \triangleright (\mathcal{X} \oplus \mathcal{Y})$ and $a \neq \hat{\varsigma}(\mathcal{X}) + \hat{\varsigma}(\mathcal{Y})$.

This fact also entails that up-to substitutions is *not* compatible. Indeed, following the general theory in [23], if it would be compatible, then open bisimilarity would be closed under substitution.

Appendix E. Details on the companion

In this section, we precisely characterise the companion (C, γ) of the functor F, $F(X) = A \times X$, show how this companion yields final Mealy machines, and ultimately prove Theorem 7.5. As already mentioned in Section 7, C is given by

$$CX = \Gamma((A^\omega)^X, A^\omega). \tag{E.1}$$

We will also prove below that $\gamma : CF \Rightarrow FC$ is given by

$$\gamma_X(\mathfrak{c}\colon (A^\omega)^{A\times X} \to A^\omega) = (\mathsf{hd}(\mathfrak{c}((a,x) \mapsto a.\alpha)), \mathfrak{c}') \tag{E.2}$$

where $\alpha \in A^\omega$ is an arbitrary stream, and $\mathfrak{c}' : (A^\omega)^{\mathcal{V}} \to A^\omega$ is given by

$$\mathfrak{c}'(\psi : X \to A^\omega) = \mathsf{tl}(\mathfrak{c}((a,x) \mapsto a.\psi(x))). \tag{E.3}$$

In the proof that (C, γ) is the companion, we will use that the (C, γ) 'constructs final Mealy machines' (Proposition A.13). First, we recall the following characterisation of final Mealy machines for $F^{A^{\mathcal{V}}}$ (which is well-known; see, e.g., [18]).

Lemma A.12. *The final $(A \times -)^{A^{\mathcal{V}}}$-coalgebra $\zeta : \tilde{\Gamma} \to (A \times \tilde{\Gamma})^{A^{\mathcal{V}}}$ is given by*

$$\zeta(\mathfrak{c}\colon (A^\omega)^{\mathcal{V}} \to A^\omega)(\varsigma : \mathcal{V} \to A) = (\mathit{hd}(\mathfrak{c}(\mathcal{X} \mapsto \varsigma(\mathcal{X}).\alpha)), \mathfrak{c}_\varsigma) \tag{E.4}$$

where $\alpha \in A^\omega$ is an arbitrary stream, and $\mathfrak{c}_\varsigma : (A^\omega)^{\mathcal{V}} \to A^\omega$ is given by

$$\mathfrak{c}_\varsigma(\psi : \mathcal{V} \to A^\omega) = \mathit{tl}(\mathfrak{c}(\mathcal{X} \mapsto \varsigma(\mathcal{X}).\psi(\mathcal{X}))). \tag{E.5}$$

Proposition A.13. *The X component of the $\mathbb{F}$-coalgebra $\widehat{\gamma}$, corresponding to the distributive law γ given in (E.2), is the final $(A \times -)^{A^X}$-coalgebra.*

Proof. This follows from a simple computation:

$$
\begin{aligned}
&\widehat{\gamma}_X(\mathfrak{c}\colon (A^\omega)^X \to A^\omega)(\varsigma\colon X \to A)\\
&= \{\text{by definition of } \widehat{\gamma}\}\\
&\gamma_X^{A^X}(cs^C_{A^X, A\times X}(C(c_X)(\mathfrak{c})))(\varsigma)\\
&= \{\text{by definition of the functor } (-)^{A^X}\}\\
&\gamma_X(cs^C_{A^X, A\times X}(C(c_X)(\mathfrak{c}))(\varsigma))\\
&= \{\text{by definition of costrength}\}\\
&\gamma_X(C(\epsilon^\varsigma \circ c_X)(\mathfrak{c}))\\
&= \{\text{by definition of } \epsilon\}\\
&\gamma_X(C(x \mapsto (\varsigma(x), x))(\mathfrak{c}))\\
&= \{\text{by definition of the functor } C \text{ on morphisms}\}\\
&\gamma_X((\varphi\colon A \times X \to A^\omega) \mapsto \mathfrak{c}(x \mapsto (\varsigma(x), x)))
\end{aligned}
$$

But after spelling out γ_X, it is immediately clear that this is exactly the Mealy machine given in Lemma A.12. □

Now, we can prove that (C, γ) as presented above is really the companion.

Proposition A.14. *The companion of $FX = A \times X$ is given by (C, γ) as defined in (E.1) and (E.2) in the beginning of this section.*

Moreover, given a distributive law $\rho\colon TF \Longrightarrow FT$, the natural transformation $\kappa\colon T \Longrightarrow C$, which is the unique morphism from (Σ, λ) to the companion (C, γ), is given by

$$\kappa_X(t \in TX)(\psi\colon \mathcal{V} \to A^\omega) = [\![(T\psi)(t)]\!]_a\,. \tag{E.6}$$

Proof. It is not very difficult to verify that $\kappa_X(t)$ is a causal function.

In order to show that (C, γ) is the companion, we first show that κ is a morphism from (T, ρ) to (C, γ), i.e. that $F\kappa \circ \rho = \gamma \circ \kappa F$. First observe that

$$\gamma(\kappa_{FX}(t \in TFX)) = (\mathsf{hd}([\![T((a, x) \mapsto a.\alpha)(t)]\!]_a), \mathfrak{c}'),$$

where $\mathfrak{c}'\colon (A^\omega)^X \to A^\omega$ is given by

$$\mathfrak{c}'(\psi\colon X \to A^\omega) = \mathsf{tl}([\![T((a, x) \mapsto a.\psi(x))(t)]\!]_a).$$

Now for the left component we have

$$
\begin{aligned}
&\mathsf{hd}([\![T((a, x) \mapsto a.\alpha)(t)]\!]_a)\\
&= \{\text{as } \mathsf{hd} = \pi_1 \circ \zeta\}\\
&\pi_1(\zeta([\![T((a, x) \mapsto a.\alpha)(t)]\!]_a))\\
&= \{\text{by definition of } [\![-]\!]_a\}\\
&\pi_1(F([\![-]\!]_a)(\rho_{A^\omega}(T(\zeta)(T((a, x) \mapsto a.\alpha)(t)))))\\
&= \{\text{using } \pi_1 \circ Ff = \pi_1\}\\
&\pi_1(\rho_{A^\omega}(T(\zeta)(T((a, x) \mapsto a.\alpha)(t))))\\
&= \{\text{as } \zeta(a.\alpha) = (a, \alpha)\}\\
&\pi_1(\rho_{A^\omega}(T((a, x) \mapsto (a, \alpha))(t)))\\
&= \{\text{by definition of the functor } F \text{ on morphisms}\}\\
&\pi_1(\rho_{A^\omega}(TF(x \mapsto \alpha)(t)))
\end{aligned}
$$

$$
\begin{aligned}
&= \{\text{by naturality of } \rho\}\\
&\pi_1(FT(x \mapsto \alpha)(\rho_X(t)))\\
&= \{\text{using } \pi_1 \circ Ff = \pi_1\}\\
&\pi_1(\rho_X(t))
\end{aligned}
$$

and for the right component we have

$$
\begin{aligned}
&\mathfrak{c}'(\psi : X \to A^\omega)\\
&= \{\text{by definition of } \mathfrak{c}'\}\\
&\mathsf{tl}(\llbracket T((a,x) \mapsto a.\psi(x))(t) \rrbracket_a)\\
&= \{\text{as } \mathsf{tl} = \pi_2 \circ \zeta\}\\
&\pi_2(\zeta(\llbracket T((a,x) \mapsto a.\psi(x))(t) \rrbracket_a))\\
&= \{\text{by definition of } \llbracket - \rrbracket_a\}\\
&\pi_2(F(\llbracket - \rrbracket_a)(\rho_{A^\omega}(T(\zeta)(T((a,x) \mapsto a.\psi(x))(t)))))\\
&= \{\text{as } \zeta(a.\psi(x)) = (a, \psi(x))\}\\
&\pi_2(F(\llbracket - \rrbracket_a)(\rho_{A^\omega}(T((a,x) \mapsto (a, \psi(x)))(t))))\\
&= \{\text{by definition of the functor } F \text{ on morphisms}\}\\
&\pi_2(F(\llbracket - \rrbracket_a)(\rho_{A^\omega}(TF(\psi)(t))))\\
&= \{\text{by naturality of } \rho\}\\
&\pi_2(F(\llbracket - \rrbracket_a)((FT(\psi)(\rho_X(t)))))\\
&= \{\text{using } \pi_2 \circ Ff = f \circ \pi_2\}\\
&\llbracket T(\psi)(\pi_2(\rho_X(t))) \rrbracket_a\\
&= \{\text{by definition of } \kappa\}\\
&\kappa_X(\pi_2(\rho_X(t)))(\psi)
\end{aligned}
$$

Thus, combining both calculations, we get $F(\kappa_X)(\rho_X(t)) = \gamma_X(\kappa_{FX}(t))$.

Finally we show that κ is also the unique such morphism. Let θ be a morphism from (T, ρ) to (C, γ). Then both θ and κ are coalgebra morphisms from $(T, \widehat{\rho})$ to $(C, \widehat{\gamma})$. And for a set X, θ_X and κ_X are coalgebra morphisms from $(TX, \widehat{\rho}_X)$ to $(CX, \widehat{\gamma}_X)$, but $(CX, \widehat{\gamma}_X)$ is the final Mealy machine according to Proposition A.13, so $\kappa_X = \theta_X$ for all sets X and thus $\kappa = \theta$. □

Putting everything together, we obtain the proof of Theorem 7.5.

Proof. [Proof of Theorem 7.5] For any $t \in T\mathcal{V}$ and $\psi : \mathcal{V} \to A^\omega$:

$$\kappa_{\mathcal{V}}(t)(\psi) = \llbracket (T\psi)(t) \rrbracket_a = \llbracket t \rrbracket_o(\psi)$$

by Proposition A.14 and Proposition 5.5, and thus $\kappa_{\mathcal{V}} = \llbracket - \rrbracket_o$. □

References

[1] R. de Simone, Higher-level synchronising devices in Meije-SCCS, Theor. Comput. Sci. 37 (1985) 245–267, https://doi.org/10.1016/0304-3975(85)90093-3.
[2] A. Rensink, Bisimilarity of open terms, Inf. Comput. 156 (1–2) (2000) 345–385, https://doi.org/10.1006/inco.1999.2818.
[3] P. Baldan, A. Bracciali, R. Bruni, A semantic framework for open processes, Theor. Comput. Sci. 389 (3) (2007) 446–483, https://doi.org/10.1016/j.tcs.2007.09.004.
[4] L. Aceto, M. Cimini, A. Ingólfsdóttir, Proving the validity of equations in GSOS languages using rule-matching bisimilarity, Math. Struct. Comput. Sci. 22 (2) (2012) 291–331, https://doi.org/10.1017/S0960129511000417.
[5] D. Lucanu, E. Goriac, G. Caltais, G. Rosu, CIRC: a behavioral verification tool based on circular coinduction, in: Proc. CALCO 2009, 2009, pp. 433–442.
[6] A. Popescu, E.L. Gunter, Incremental pattern-based coinduction for process algebra and its Isabelle formalization, in: Proc. FOSSACS, 2010, pp. 109–127.
[7] H. Zantema, J. Endrullis, Proving equality of streams automatically, in: Proceedings of RTA 2011, Novi Sad, Serbia, 2011, pp. 393–408.
[8] B. Klin, Bialgebras for structural operational semantics: an introduction, Theor. Comput. Sci. 412 (38) (2011) 5043–5069, https://doi.org/10.1016/j.tcs.2011.03.023.
[9] J. Rutten, Universal coalgebra: a theory of systems, Theor. Comput. Sci. 249 (1) (2000) 3–80, https://doi.org/10.1016/S0304-3975(00)00056-6.

[10] J. Rutten, A tutorial on coinductive stream calculus and signal flow graphs, Theor. Comput. Sci. 343 (3) (2005) 443–481, https://doi.org/10.1016/j.tcs.2005.06.019.
[11] H.H. Hansen, B. Klin, Pointwise extensions of GSOS-defined operations, Math. Struct. Comput. Sci. 21 (2) (2011) 321–361, https://doi.org/10.1017/S096012951000054X.
[12] D. Pous, D. Sangiorgi, Enhancements of the bisimulation proof method, in: D. Sangiorgi, J. Rutten (Eds.), Advanced Topics in Bisimulation and Coinduction, Cambridge University Press, 2011, pp. 233–289.
[13] H.H. Hansen, C. Kupke, J. Rutten, Stream differential equations: specification formats and solution methods, Logical Methods in Computer Science 13 (1), https://doi.org/10.23638/LMCS-13(1:3)2017.
[14] F. Bonchi, M.D. Lee, J. Rot, Bisimilarity of open terms in stream GSOS, in: M. Dastani, M. Sirjani (Eds.), Fundamentals of Software Engineering - 7th International Conference, FSEN 2017, Tehran, Iran, April 26–28, 2017, in: Lecture Notes in Computer Science, vol. 10522, Springer, 2017, pp. 35–50, Revised Selected Papers.
[15] D. Pous, J. Rot, Companions, codensity and causality, in: Proc. FOSSACS, 2017, pp. 106–123.
[16] H. Basold, D. Pous, J. Rot, Monoidal company for accessible functors, in: 7th Conference on Algebra and Coalgebra in Computer Science, CALCO 2017, Ljubljana, Slovenia, June 12–16, 2017, chap. 5, 16 pp.
[17] D. Turi, G.D. Plotkin, Towards a mathematical operational semantics, in: Proceedings LICS 1997, 1997, pp. 280–291.
[18] H.H. Hansen, J.J.M.M. Rutten, Symbolic synthesis of mealy machines from arithmetic bitstream functions, Sci. Ann. Comput. Sci. 20 (2010) 97–130.
[19] M. Mousavi, M. Reniers, J. Groote, SOS formats and meta-theory: 20 years after, Theor. Comput. Sci. 373 (3) (2007) 238–272, https://doi.org/10.1016/j.tcs.2006.12.019.
[20] J. Rutten, Elements of stream calculus (an extensive exercise in coinduction), Electron. Notes Theor. Comput. Sci. 45 (2001) 358–423, https://doi.org/10.1016/S1571-0661(04)80972-1.
[21] B. Bloom, S. Istrail, A.R. Meyer, Bisimulation can't be traced, J. ACM 42 (1) (1995) 232–268, https://doi.org/10.1145/200836.200876.
[22] F. Bartels, On Generalised Coinduction and Probabilistic Specification Formats, Ph.D. thesis, CWI, Amsterdam, 2004.
[23] F. Bonchi, D. Petrisan, D. Pous, J. Rot, A general account of coinduction up-to, Acta Inform. 54 (2) (2017) 127–190, https://doi.org/10.1007/s00236-016-0271-4.
[24] B. Klin, B. Nachyla, Presenting morphisms of distributive laws, in: 6th Conference on Algebra and Coalgebra in Computer Science, CALCO, 2015, June 24–26, 2015, Nijmegen, The Netherlands, 2015, pp. 190–204.
[25] M. Lenisa, J. Power, H. Watanabe, Distributivity for endofunctors, pointed and co-pointed endofunctors, monads and comonads, Electron. Notes Theor. Comput. Sci. 33 (2000) 230–260, https://doi.org/10.1016/S1571-0661(05)80350-0.
[26] J. Power, H. Watanabe, Combining a monad and a comonad, Theor. Comput. Sci. 280 (1–2) (2002) 137–162, https://doi.org/10.1016/S0304-3975(01)00024-X.
[27] H. Watanabe, Well-behaved translations between structural operational semantics, Electron. Notes Theor. Comput. Sci. 65 (1) (2002) 337–357, https://doi.org/10.1016/S1571-0661(04)80372-4.
[28] J. Endrullis, D. Hendriks, M. Bodin, Circular coinduction in Coq using bisimulation-up-to techniques, in: S. Blazy, C. Paulin-Mohring, D. Pichardie (Eds.), Interactive Theorem Proving – 4th International Conference, ITP 2013, Rennes, France, July 22–26, 2013, Proceedings, in: Lecture Notes in Computer Science, vol. 7998, Springer, 2013, pp. 354–369.
[29] A. Silva, Kleene Coalgebra, [Sl: sn], 2010.
[30] R. Milner, A complete inference system for a class of regular behaviours, J. Comput. Syst. Sci. 28 (3) (1984) 439–466.
[31] L. Aceto, W. Fokkink, A. Ingolfsdottir, B. Luttik, Finite equational bases in process algebra: results and open questions, in: Processes, Terms and Cycles: Steps on the Road to Infinity, Springer, 2005, pp. 338–367.
[32] J.F. Groote, F. Vaandrager, Structured operational semantics and bisimulation as a congruence, Inf. Comput. 100 (2) (1992) 202–260.
[33] J.F. Groote, Transition system specifications with negative premises, Theor. Comput. Sci. 118 (2) (1993) 263–299.
[34] B. Jacobs, A bialgebraic review of deterministic automata, regular expressions and languages, in: Algebra, Meaning, and Computation, 2006, pp. 375–404.