Information and Computation 209 (2011) 822–849

Contents lists available at ScienceDirect

Information and Computation

journal homepage: www.elsevier.com/locate/ic

Quantitative Kleene coalgebras

Alexandra Silva [b,*], Filippo Bonchi [a], Marcello Bonsangue [c,b], Jan Rutten [b,e,d]

[a] CNRS, ENS-Lyon, 7 Passage Vercors, 69007 Lyon, France
[b] Centrum Wiskunde & Informatica, Science Park 123, 1098 XG Amsterdam, The Netherlands
[c] Leiden Institute Advanced Computer Science, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
[d] Radboud Universiteit Nijmegen, Heyendaalseweg 135, 6525 AJ Nijmegen, The Netherlands
[e] Vrije Universiteit Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands

ARTICLE INFO

Article history:
Received 29 January 2010
Revised 3 August 2010
Available online 21 December 2010

ABSTRACT

We present a systematic way to generate (1) languages of (generalized) regular expressions, and (2) sound and complete axiomatizations thereof, for a wide variety of quantitative systems. Our quantitative systems include weighted versions of automata and transition systems, in which transitions are assigned a value in a monoid that represents cost, duration, probability, etc. Such systems are represented as coalgebras and (1) and (2) above are derived in a modular fashion from the underlying (functor) type of these coalgebras.

In previous work, we applied a similar approach to a class of systems (without weights) that generalizes both the results of Kleene (on rational languages and DFA's) and Milner (on regular behaviours and finite LTS's), and includes many other systems such as Mealy and Moore machines.

In the present paper, we extend this framework to deal with quantitative systems. As a consequence, our results now include languages and axiomatizations, both existing and new ones, for many different kinds of probabilistic systems.

1. Introduction

Kleene's Theorem [25] gives a fundamental correspondence between *regular expressions* and *deterministic finite automata* (DFA's): each regular expression denotes a language that can be recognized by a DFA and, vice-versa, the language accepted by a DFA can be specified by a regular expression. Languages denoted by regular expressions are called *regular*. Two regular expressions are called (language) equivalent if they denote the same regular language. Salomaa [37] presented a sound and complete axiomatization for proving the equivalence of regular expressions, which was later refined by Kozen [27].

The above programme was applied by Milner [31] to process behaviours and labelled transition systems (LTS's). Milner introduced a set of expressions for finite LTS's and proved an analogue of Kleene's Theorem: each expression denotes the behaviour of a finite LTS and, conversely, the behaviour of a finite LTS can be specified by an expression (modulo bisimilarity). Milner also provided an axiomatization for his expressions, with the property that two expressions are provably equivalent if and only if they are bisimilar.

Coalgebras provide a general framework for the study of dynamical systems such as DFA's and LTS's. For a functor $\mathcal{F}$: **Set** $\to$ **Set**, an $\mathcal{F}$-coalgebra or $\mathcal{F}$-system is a pair (S, f), consisting of a set S of states and a function $f: S \to \mathcal{F}(S)$ defining the "transitions" of the states. We call the functor $\mathcal{F}$ the *type* of the system. For instance, DFA's can be readily seen to correspond to coalgebras of the functor $\mathcal{F}(S) = 2 \times S^A$ and image-finite LTS's are obtained by $\mathcal{F}(S) = \mathcal{P}_\omega(S)^A$, where $\mathcal{P}_\omega$ is finite powerset.

* Corresponding author. Fax: +31 205924199.
E-mail address: ams@cwi.nl (A. Silva).

0890-5401/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ic.2010.09.007

Under mild conditions, functors $\mathcal{F}$ have a *final coalgebra* (unique up to isomorphism) into which every $\mathcal{F}$-coalgebra can be mapped via a unique so-called $\mathcal{F}$-*homomorphism*. The final coalgebra can be viewed as the universe of all possible $\mathcal{F}$-*behaviours*: the unique homomorphism into the final coalgebra maps every state of a coalgebra to a canonical representative of its behaviour. This gives a general notion of behavioural equivalence: two states are equivalent iff they are mapped to the same element of the final coalgebra. In the case of DFA's, two states are equivalent when they accept the same language; for LTS's, behavioural equivalence coincides with the ordinary notion of bisimilarity.

In a previous paper [8], we introduced for coalgebras of a large but restricted class of functors, a language of regular expressions; a corresponding generalization of Kleene's Theorem; and a sound and complete axiomatization with respect to bisimilarity. We derived both the language of expressions and their axiomatization, in a modular fashion, from the functor defining the type of the system, by induction on the structure of the functors.

In recent years, much attention has been devoted to the analysis of probabilistic behaviours, which occur for instance in randomized, fault-tolerant systems. Several different types of systems were proposed: reactive [28,34], generative [19], stratified [41,45], alternating [23,46], (simple) Segala systems [39,40], bundle [15] and Pnueli–Zuck [33], amongst others. For some of these systems, expressions were defined for the specification of their behaviours, as well as axioms to reason about their behavioural equivalence. Examples include [1,2,4,16,17,24,29,32,42].

The results in [8] apply to the class of non-deterministic functors, which is general enough to include the examples of deterministic automata and labelled transition systems, as well as many other systems such as Mealy and Moore machines. However, probabilistic systems, weighted automata [18,38], etc. *cannot* be described by non-deterministic functors. It is aim of the present paper to identify a class of functors (a) that is general enough to include these and more generally a large class of *quantitative systems*; and (b) to which the methodology developed in [8] can be extended.

To this end, we give a non-trivial extension of the class of non-deterministic functors by adding a functor type that allows the transitions of our systems to take values in a *monoid* structure of quantitative values. This new class, which we shall call quantitative functors, now includes all the types of probabilistic systems mentioned above.

At the same time, we show how to extend our earlier approach to the new setting. As it turns out, the main technical challenge is due to the fact that the behaviour of quantitative systems is inherently *non-idempotent*. As an example consider the expression $1/2 \cdot \varepsilon \oplus 1/2 \cdot \varepsilon'$ representing a probabilistic system that either behaves as ε with probability $1/2$ or behaves as ε' with the same probability. When ε is equivalent to ε', then the system is equivalent to $1 \cdot \varepsilon$ rather than $1/2 \cdot \varepsilon$. This is problematic because idempotency played a crucial role in our previous results to ensure that expressions denote finite-state behaviours.

We will show how the lack of idempotency in the extended class of functors can be circumvented by a clever use of the monoid structure. This will allow us to derive for each functor in our new extended class everything we were after: a language of regular expressions; a corresponding Kleene's Theorem; and a sound and complete axiomatization for the corresponding notion of behavioural equivalence.

In order to show the effectiveness and the generality of our approach, we apply it to four types of systems: weighted automata; simple Segala, stratified and Pnueli–Zuck systems. For simple Segala systems, we recover the language and axiomatization presented in [17]. For weighted automata and stratified systems, languages have been defined in [12] and [45] but, to the best of our knowledge, no axiomatization was ever given. Applying our method, we obtain the same languages and, more interestingly, we obtain novel axiomatizations. We also present a completely new framework to reason about Pnueli–Zuck systems. Table 1 summarizes the main results of this paper: the languages and axiomatizations derived for several quantitative systems.

This paper is an extended version of our CONCUR paper [6]. In comparison to [6], this paper includes more details in the examples, contains all the proofs of new results, provides a detailed explanation of the soundness and completeness of the axiomatization and it includes the description of an alternative definition of a functor to model quantitative systems (Section 7).

Organization of the paper. Section 2 gives preliminaries on coalgebras and non-deterministic functors and recalls the main results of [8]. In Section 3, we introduce the functor that will allow us to model quantitative systems: the monoidal exponentiation functor. Section 4 shows how to extend the framework presented in the previous chapter to quantitative systems: we associate with every quantitative functor $\mathcal{H}$ a language of expressions $\mathrm{Exp}_{\mathcal{H}}$, we prove a Kleene like theorem and we introduce a sound and complete axiomatization with respect to behavioural equivalence of $\mathcal{H}$. Section 5 paves the way for the derivation of expressions and axioms for probabilistic systems, which we present in Section 6. Section 7 shows a variation on the definition of the monoidal exponentiation functor and the consequences it would have in the framework. We conclude and present pointers for future work in Section 8.

2. Preliminaries

We give the basic definitions on non-deterministic functors and coalgebras and introduce the notion of bisimulation.

First we fix notation on sets and operations on them. Let **Set** be the category of sets and functions. Sets are denoted by capital letters $X, Y, \ldots$ and functions by lower case $f, g, \ldots$ We write $\{\}$ for the empty set and the collection of all *finite* subsets of a set X is defined as $\mathcal{P}_\omega(X) = \{Y \subseteq X \mid Y \text{ finite}\}$. The collection of functions from a set X to a set Y is denoted by

Table 1
All the (valid) expressions are closed and guarded. The congruence and the α-equivalence axioms are implicitly assumed for all the systems. The symbols 0 and $+$ denote, in the case of weighted automata, the empty element and the binary operator of the commutative monoid $\mathbb{S}$ whilst, for the other systems, they denote the ordinary 0 and sum of real numbers. We write $\bigoplus_{i\in 1,\dots,n} p_i \cdot \varepsilon_i$ for $p_1 \cdot \varepsilon_1 \oplus \cdots \oplus p_n \cdot \varepsilon_n$.

Weighted automata – $\mathcal{H}(S) = \mathbb{S} \times (\mathbb{S}^S)^A$

$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \oplus \varepsilon \mid \mu x.\varepsilon \mid x \mid s \mid a(s \cdot \varepsilon)$ where $s \in \mathbb{S}$ and $a \in A$

$(\varepsilon_1 \oplus \varepsilon_2) \oplus \varepsilon_3 \equiv \varepsilon_1 \oplus (\varepsilon_2 \oplus \varepsilon_3)$ $\quad \varepsilon_1 \oplus \varepsilon_2 \equiv \varepsilon_2 \oplus \varepsilon_1$ $\quad \varepsilon \oplus \underline{\emptyset} \equiv \varepsilon$
$a(s \cdot \varepsilon) \oplus a(s' \cdot \varepsilon) \equiv a((s + s') \cdot \varepsilon)$ $\quad s \oplus s' \equiv s + s'$ $\quad a(0 \cdot \varepsilon) \equiv \underline{\emptyset}$ $\quad 0 \equiv \underline{\emptyset}$
$\varepsilon[\mu x.\varepsilon/x] \equiv \mu x.\varepsilon$ $\quad \gamma[\varepsilon/x] \equiv \varepsilon \Rightarrow \mu x.\gamma \equiv \varepsilon$

Segala systems – $\mathcal{H}(S) = \mathcal{P}_\omega(\mathcal{D}_\omega(S))^A$

$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \boxplus \varepsilon \mid \mu x.\varepsilon \mid x \mid a(\{\varepsilon'\})$ where $a \in A$, $p_i \in (0, 1]$ and $\sum_{i\in 1,\dots,n} p_i = 1$
$\varepsilon' ::= \bigoplus_{i\in 1,\dots,n} p_i \cdot \varepsilon_i$

$(\varepsilon_1 \boxplus \varepsilon_2) \boxplus \varepsilon_3 \equiv \varepsilon_1 \boxplus (\varepsilon_2 \boxplus \varepsilon_3)$ $\quad \varepsilon_1 \boxplus \varepsilon_2 \equiv \varepsilon_2 \boxplus \varepsilon_1$ $\quad \varepsilon \boxplus \underline{\emptyset} \equiv \varepsilon$ $\quad \varepsilon \boxplus \varepsilon \equiv \varepsilon$
$(\varepsilon'_1 \oplus \varepsilon'_2) \oplus \varepsilon'_3 \equiv \varepsilon'_1 \oplus (\varepsilon'_2 \oplus \varepsilon'_3)$ $\quad \varepsilon'_1 \oplus \varepsilon'_2 \equiv \varepsilon'_2 \oplus \varepsilon'_1$ $\quad (p_1 \cdot \varepsilon) \oplus (p_2 \cdot \varepsilon) \equiv (p_1 + p_2) \cdot \varepsilon$
$\varepsilon[\mu x.\varepsilon/x] \equiv \mu x.\varepsilon$ $\quad \gamma[\varepsilon/x] \equiv \varepsilon \Rightarrow \mu x.\gamma \equiv \varepsilon$

Stratified systems – $\mathcal{H}(S) = \mathcal{D}_\omega(S) + (\mathrm{B} \times S) + 1$

$\varepsilon ::= \mu x.\varepsilon \mid x \mid \langle b, \varepsilon \rangle \mid \bigoplus_{i\in 1,\dots,n} p_i \cdot \varepsilon_i \mid \downarrow$ where $b \in \mathrm{B}$, $p_i \in (0, 1]$ and $\sum_{i\in 1,\dots,n} p_i = 1$

$(\varepsilon_1 \oplus \varepsilon_2) \oplus \varepsilon_3 \equiv \varepsilon_1 \oplus (\varepsilon_2 \oplus \varepsilon_3)$ $\quad \varepsilon_1 \oplus \varepsilon_2 \equiv \varepsilon_2 \oplus \varepsilon_1$ $\quad (p_1 \cdot \varepsilon) \oplus (p_2 \cdot \varepsilon) \equiv (p_1 + p_2) \cdot \varepsilon$
$\varepsilon[\mu x.\varepsilon/x] \equiv \mu x.\varepsilon$ $\quad \gamma[\varepsilon/x] \equiv \varepsilon \Rightarrow \mu x.\gamma \equiv \varepsilon$

Pnueli–Zuck systems – $\mathcal{H}(S) = \mathcal{P}_\omega\mathcal{D}_\omega(\mathcal{P}_\omega(S)^A)$

$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \boxplus \varepsilon \mid \mu x.\varepsilon \mid x \mid \{\varepsilon'\}$ where $a \in A$, $p_i \in (0, 1]$ and $\sum_{i\in 1,\dots,n} p_i = 1$
$\varepsilon' ::= \bigoplus_{i\in 1,\dots,n} p_i \cdot \varepsilon''_i$
$\varepsilon'' ::= \underline{\emptyset} \mid \varepsilon'' \boxplus \varepsilon'' \mid a(\{\varepsilon\})$

$(\varepsilon_1 \boxplus \varepsilon_2) \boxplus \varepsilon_3 \equiv \varepsilon_1 \boxplus (\varepsilon_2 \boxplus \varepsilon_3)$ $\quad \varepsilon_1 \boxplus \varepsilon_2 \equiv \varepsilon_2 \boxplus \varepsilon_1$ $\quad \varepsilon \boxplus \underline{\emptyset} \equiv \varepsilon$ $\quad \varepsilon \boxplus \varepsilon \equiv \varepsilon$
$(\varepsilon'_1 \oplus \varepsilon'_2) \oplus \varepsilon'_3 \equiv \varepsilon'_1 \oplus (\varepsilon'_2 \oplus \varepsilon'_3)$ $\quad \varepsilon'_1 \oplus \varepsilon'_2 \equiv \varepsilon'_2 \oplus \varepsilon'_1$ $\quad (p_1 \cdot \varepsilon'') \oplus (p_2 \cdot \varepsilon'') \equiv (p_1 + p_2) \cdot \varepsilon''$
$(\varepsilon''_1 \boxplus \varepsilon''_2) \boxplus \varepsilon''_3 \equiv \varepsilon''_1 \boxplus (\varepsilon''_2 \boxplus \varepsilon''_3)$ $\quad \varepsilon''_1 \boxplus \varepsilon''_2 \equiv \varepsilon''_2 \boxplus \varepsilon''_1$ $\quad \varepsilon'' \boxplus \underline{\emptyset} \equiv \varepsilon''$ $\quad \varepsilon'' \boxplus \varepsilon'' \equiv \varepsilon''$
$\varepsilon[\mu x.\varepsilon/x] \equiv \mu x.\varepsilon$ $\quad \gamma[\varepsilon/x] \equiv \varepsilon \Rightarrow \mu x.\gamma \equiv \varepsilon$

Y^X. We write id_X for the identity function on set X. Given functions $f : X \to Y$ and $g : Y \to Z$ we write their composition as $g \circ f$. The product of two sets X, Y is written as $X \times Y$, with projection functions $X \xleftarrow{\pi_1} X \times Y \xrightarrow{\pi_2} Y$. The set 1 is a singleton set typically written as $1 = \{*\}$ and it can be regarded as the empty product. We define

$$X \oplus Y = X \uplus Y \uplus \{\bot, \top\}$$

where $\uplus$ is the disjoint union of sets, with injections $X \xrightarrow{\kappa_1} X \uplus Y \xleftarrow{\kappa_2} Y$. Note that the set $X \oplus Y$ is different from the classical coproduct of X and Y (which we shall denote by $X + Y$), because of the two extra elements $\bot$ and $\top$. These extra elements will later be used to represent, respectively, underspecification and inconsistency in the specification of some systems.

For each of the operations defined above on sets, there are analogous ones on functions. Let $f_1 : X \to Y$ and $f_2 : Z \to W$. We define the following operations:

$$f_1 \times f_2 : X \times Z \to Y \times W \qquad f_1 \oplus f_2 : X \oplus Z \to Y \oplus W$$
$$(f_1 \times f_2)(\langle x, z \rangle) = \langle f_1(x), f_2(z) \rangle \qquad (f \oplus f_2)(c) = c,\ c \in \{\bot, \top\}$$
$$(f_1 \oplus f_2)(\kappa_i(x)) = \kappa_i(f_i(x)),\ i \in \{1, 2\}$$
$$f_1^A : X^A \to Y^A \qquad \mathcal{P}_\omega(f_1) : \mathcal{P}_\omega(X) \to \mathcal{P}_\omega(Y)$$
$$f_1^A(g) = \lambda a.f_1(g(a)) \qquad \mathcal{P}_\omega(f_1)(S) = \{f_1(x) \mid x \in S\}$$

Note that here we are using the same symbols that we defined above for the operations on sets. It will always be clear from the context which operation is being used.

A functor $\mathcal{F}$: **Set** $\to$ **Set** is a mapping of sets to sets and functions to functions satisfying:

1. $\mathcal{F}(g \circ f) = \mathcal{F}(g) \circ \mathcal{F}(f)$
2. $\mathcal{F}(id_X) = id_{\mathcal{F}(X)}$

The operations we defined above on sets and functions actually form functors, as we will use later in this paper when defining the class of quantitative functors.

In our definition of quantitative functors we will use constant sets equipped with an information order. In particular, we will use join-semilattices. A (bounded) join-semilattice is a set B equipped with a binary operation $\vee_{\mathrm{B}}$ and a constant $\bot_{\mathrm{B}} \in \mathrm{B}$, such that $\vee_{\mathrm{B}}$ is commutative, associative and idempotent. The element $\bot_{\mathrm{B}}$ is neutral w.r.t. $\vee_{\mathrm{B}}$. As usual, $\vee_{\mathrm{B}}$ gives rise to a partial ordering $\leq_{\mathrm{B}}$ on the elements of B: $b_1 \leq_{\mathrm{B}} b_2 \Leftrightarrow b_1 \vee_{\mathrm{B}} b_2 = b_2$. Every set S can be mapped into a join-semilattice by taking B to be the set of all finite subsets of S with union as join.

Coalgebras. A coalgebra is a pair $(S, f : S \to \mathcal{F}(S))$, where S is a set of states and $\mathcal{F}$: **Set** $\to$ **Set** is a functor. The functor $\mathcal{F}$, together with the function f, determines the *transition structure* (or dynamics) of the $\mathcal{F}$-coalgebra [35].

An $\mathcal{F}$-*homomorphism* from an $\mathcal{F}$-coalgebra (S, f) to an $\mathcal{F}$-coalgebra (T, g) is a function $h : S \to T$ preserving the transition structure, *i.e.*, such that $g \circ h = \mathcal{F}(h) \circ f$.

Definition 2.1. An $\mathcal{F}$-coalgebra (Ω, ω) is said to be *final* if for any $\mathcal{F}$-coalgebra (S, f) there exists a unique $\mathcal{F}$-homomorphism $\mathbf{beh}_S : S \to \Omega$.

The notion of finality will play a key role later in providing a semantics to expressions.

For every bounded functor there exists a final $\mathcal{F}$-coalgebra $(\Omega_{\mathcal{F}}, \omega_{\mathcal{F}})$ [22,35]. A functor is said to be bounded [22, Theorem 4.7] if there exists a natural surjection η from a functor $B \times (-)^A$ to $\mathcal{F}$, for some sets B and A.

Given an $\mathcal{F}$-coalgebra (S, f) and a subset V of S with inclusion map $i : V \to S$ we say that V is a subcoalgebra of S if there exists $g : V \to \mathcal{F}(V)$ such that i is an $\mathcal{F}$-homomorphism. Given $s \in S$, $\langle s \rangle = (T, t)$ denotes the smallest subcoalgebra generated by s, with T given by

$$T = \bigcap \{V \mid V \text{ is a subcoalgebra of } S \text{ and } s \in V\} \tag{1}$$

If the functor $\mathcal{F}$ preserves arbitrary intersections, then the subcoalgebra $\langle s \rangle$ exists. This will be the case for every functor considered in this paper.

We will write $Coalg(\mathcal{F})$ for the category of $\mathcal{F}$-coalgebras together with coalgebra homomorphisms. We also write $Coalg_{\mathbf{LF}}(\mathcal{F})$ for the category of $\mathcal{F}$-coalgebras that are *locally finite*: $\mathcal{F}$-coalgebras (S, f) such that for each state $s \in S$ the generated subcoalgebra $\langle s \rangle$ is finite.

Let (S, f) and (T, g) be two $\mathcal{F}$-coalgebras. We call a relation $R \subseteq S \times T$ a *bisimulation* iff

$$\langle s, t \rangle \in R \Rightarrow \langle f(s), g(t) \rangle \in \overline{\mathcal{F}}(R)$$

where $\overline{\mathcal{F}}(R)$ is defined as $\overline{\mathcal{F}}(R) = \{\langle \mathcal{F}(\pi_1)(x), \mathcal{F}(\pi_2)(x) \rangle \mid x \in \mathcal{F}(R)\}$.

We write $s \sim_{\mathcal{F}} t$ whenever there exists a bisimulation relation containing (s, t) and we call $\sim_{\mathcal{F}}$ the bisimilarity relation. We shall drop the subscript $\mathcal{F}$ whenever the functor $\mathcal{F}$ is clear from the context.

We say that the states $s \in S$ and $t \in T$ are *behaviourally equivalent*, written $s \sim_b t$, if and only if they are mapped into the same element in the final coalgebra, that is $\mathbf{beh}_S(s) = \mathbf{beh}_T(t)$.

If two states are bisimilar then they are always behaviourally equivalent ($s \sim t \Rightarrow s \sim_b t$). The converse implication is only true for certain classes of functors. For instance, if the functor $\mathcal{F}$ preserves weak-pullbacks then we also have $s \sim_b t \Rightarrow s \sim t$. The class of non-deterministic functors, which we will recall next, satisfies this property, whereas the class of quantitative functors, which we shall introduce later in this paper does not.

Non-deterministic functors. Non-deterministic functors are functors $\mathcal{G}$: **Set** $\to$ **Set**, built inductively from the identity and constants, using $\times$, $\oplus$, $(-)^A$ and $\mathcal{P}_\omega$.

Definition 2.2. The class *NDF* of *non-deterministic functors* on **Set** is inductively defined by:

$$NDF \ni \mathcal{G} ::= \mathrm{Id} \mid \mathrm{B} \mid \mathcal{G} \times \mathcal{G} \mid \mathcal{G} \oplus \mathcal{G} \mid \mathcal{G}^A \mid \mathcal{P}_\omega \mathcal{G}$$

where B is a finite join-semilattice and A is a finite set.

Since we only consider finite exponents $A = \{a_1, \ldots, a_n\}$, the functor $(-)^A$ is not really needed, since it is subsumed by a product with n components. However, to simplify the presentation, we decided to include it.

We now show the explicit definition of the functors above on a set X and on a morphism $f : X \to Y$ (note that $\mathcal{G}(f) : \mathcal{G}(X) \to \mathcal{G}(Y)$).

$$\begin{array}{lll} \mathrm{Id}(X) = X & \mathrm{B}(X) = \mathrm{B} & (\mathcal{G}_1 \oplus \mathcal{G}_2)(X) = \mathcal{G}_1(X) \oplus \mathcal{G}_2(X) \\ \mathrm{Id}(f) = f & \mathrm{B}(f) = id_{\mathrm{B}} & (\mathcal{G}_1 \oplus \mathcal{G}_2)(f) = \mathcal{G}_1(f) \oplus \mathcal{G}_2(f) \end{array}$$

$$\begin{array}{lll} (\mathcal{G}^A)(X) = \mathcal{G}(X)^A & (\mathcal{P}_\omega G)(X) = \mathcal{P}_\omega(\mathcal{G}(X)) & (\mathcal{G}_1 \times \mathcal{G}_2)(X) = \mathcal{G}_1(X) \times \mathcal{G}_2(X) \\ (\mathcal{G}^A)(f) = \mathcal{G}(f)^A & (\mathcal{P}_\omega \mathcal{G})(f) = \mathcal{P}_\omega(\mathcal{G}(f)) & (\mathcal{G}_1 \times \mathcal{G}_2)(f) = \mathcal{G}_1(f) \times \mathcal{G}_2(f) \end{array}$$

Typical examples of non-deterministic functors include $\mathcal{M} = (\mathrm{B} \times \mathrm{Id})^A$, $\mathcal{D} = 2 \times \mathrm{Id}^A$, $\mathcal{Q} = (1 \oplus \mathrm{Id})^A$ and $\mathcal{N} = 2 \times (\mathcal{P}_\omega(\mathrm{Id}))^A$. These functors represent, respectively, the type of Mealy, deterministic, partial deterministic and non-deterministic automata. In [7], we have studied in detail regular expressions for Mealy automata. Similarly to what happened there, we impose a join-semilattice structure on the constant functor $\mathcal{G}(X) = \mathrm{B}$. The product, exponentiation and powerset functors preserve the join-semilattice structure and thus need not to be changed. This is not the case for the classical coproduct and thus we use $\oplus$ instead, which also guarantees that the join semilattice structure is preserved.

We remark that every non-deterministic functor is bounded (and thus has a final coalgebra).

Next, we give the definition of the ingredient relation, which relates a non-deterministic functor $\mathcal{G}$ with its *ingredients*, i.e., the functors used in its inductive construction. We shall use this relation later for typing our expressions.

Definition 2.3. Let $\lhd \subseteq \mathit{NDF} \times \mathit{NDF}$ be the least reflexive and transitive relation on non-deterministic functors such that

$$\mathcal{G}_1 \lhd \mathcal{G}_1 \times \mathcal{G}_2, \quad \mathcal{G}_2 \lhd \mathcal{G}_1 \times \mathcal{G}_2, \quad \mathcal{G}_1 \lhd \mathcal{G}_1 \oplus \mathcal{G}_2, \quad \mathcal{G}_2 \lhd \mathcal{G}_1 \oplus \mathcal{G}_2, \quad \mathcal{G} \lhd \mathcal{G}^A, \quad \mathcal{G} \lhd \mathcal{P}_\omega \mathcal{G}$$

Here and throughout this paper we use $\mathcal{F} \lhd \mathcal{G}$ as a shorthand for $\langle \mathcal{F}, \mathcal{G} \rangle \in \lhd$. If $\mathcal{F} \lhd \mathcal{G}$, then $\mathcal{F}$ is said to be an *ingredient* of $\mathcal{G}$. For example, 2, Id, Id^A and $\mathcal{D}$ itself are all the ingredients of the deterministic automata functor $\mathcal{D}$.

2.1. A language of expressions for non-deterministic coalgebras

In this section, we recall the main definitions and results concerning the language of expressions associated with a non-deterministic functor [8]. We start by introducing an untyped language of expressions and then we single out the well-typed ones via an appropriate typing system, thereby associating expressions to non-deterministic functors.

Definition 2.4 (*Expressions*)**.** Let A be a finite set, B a finite join-semilattice and X a set of fixed-point variables. The set Exp of all *expressions* is given by the following grammar, where $a \in A$, $b \in \mathrm{B}$ and $x \in X$:

$$\varepsilon ::= \underline{\emptyset} \mid x \mid \varepsilon \oplus \varepsilon \mid \mu x.\gamma \mid b \mid l\langle\varepsilon\rangle \mid r\langle\varepsilon\rangle \mid l[\varepsilon] \mid r[\varepsilon] \mid a(\varepsilon) \mid \{\varepsilon\}$$

where γ is a *guarded expression* given by:

$$\gamma ::= \underline{\emptyset} \mid \gamma \oplus \gamma \mid \mu x.\gamma \mid b \mid l\langle\varepsilon\rangle \mid r\langle\varepsilon\rangle \mid l[\varepsilon] \mid r[\varepsilon] \mid a(\varepsilon) \mid \{\varepsilon\}$$

In the expression $\mu x.\gamma$, μ is a binder for all the occurrences of x in γ. Variables that are not bound are free. A *closed expression* is an expression without free occurrences of fixed-point variables x. We denote the set of closed expressions by Exp^c.

Intuitively, expressions denote elements of the final coalgebra. The expressions $\underline{\emptyset}$, $\varepsilon_1 \oplus \varepsilon_2$ and $\mu x.\,\varepsilon$ will play a similar role to, respectively, the empty language, the union of languages and the Kleene star in classical regular expressions for deterministic automata. The expressions $l\langle\varepsilon\rangle$ and $r\langle\varepsilon\rangle$ refer to the left and right-hand side of products. Similarly, $l[\varepsilon]$ and $r[\varepsilon]$ refer to the left and right hand-side of sums. The expressions $a(\varepsilon)$ and $\{\varepsilon\}$ denote function application and a singleton set, respectively.

Next, we present a typing assignment system for associating expressions to non-deterministic functors. This will associate with each functor $\mathcal{G}$ the expressions $\varepsilon \in \mathrm{Exp}$ that are valid specifications of $\mathcal{G}$-coalgebras. The typing proceeds following the structure of the expressions and the ingredients of the functors.

Definition 2.5 (*Type system*)**.** We now define a typing relation $\vdash \subseteq \mathrm{Exp} \times \mathit{NDF} \times \mathit{NDF}$ that will associate an expression ε with two non-deterministic functors $\mathcal{F}$ and $\mathcal{G}$, which are related by the ingredient relation ($\mathcal{F}$ is an ingredient of $\mathcal{G}$). We shall write $\vdash \varepsilon : \mathcal{F} \lhd \mathcal{G}$ for $\langle \varepsilon, \mathcal{F}, \mathcal{G} \rangle \in \vdash$. The rules that define $\vdash$ are the following:

$$\frac{}{\vdash \underline{\emptyset} : \mathcal{F} \lhd \mathcal{G}} \qquad \frac{}{\vdash b : \mathrm{B} \lhd \mathcal{G}} \qquad \frac{}{\vdash x : \mathcal{G} \lhd \mathcal{G}} \qquad \frac{\vdash \varepsilon : \mathcal{G} \lhd \mathcal{G}}{\vdash \mu x.\varepsilon : \mathcal{G} \lhd \mathcal{G}}$$

$$\frac{\vdash \varepsilon_1 : \mathcal{F} \lhd \mathcal{G} \quad \vdash \varepsilon_2 : \mathcal{F} \lhd \mathcal{G}}{\vdash \varepsilon_1 \oplus \varepsilon_2 : \mathcal{F} \lhd \mathcal{G}} \qquad \frac{\vdash \varepsilon : \mathcal{G} \lhd \mathcal{G}}{\vdash \varepsilon : \mathrm{Id} \lhd \mathcal{G}} \qquad \frac{\vdash \varepsilon : \mathcal{F} \lhd \mathcal{G}}{\vdash \{\varepsilon\} : \mathcal{P}_\omega \mathcal{F} \lhd \mathcal{G}} \qquad \frac{\vdash \varepsilon : \mathcal{F} \lhd \mathcal{G}}{\vdash a(\varepsilon) : \mathcal{F}^A \lhd \mathcal{G}}$$

$$\frac{\vdash \varepsilon : \mathcal{F}_1 \lhd \mathcal{G}}{\vdash l\langle\varepsilon\rangle : \mathcal{F}_1 \times \mathcal{F}_2 \lhd \mathcal{G}} \quad \frac{\vdash \varepsilon : \mathcal{F}_2 \lhd \mathcal{G}}{\vdash r\langle\varepsilon\rangle : \mathcal{F}_1 \times \mathcal{F}_2 \lhd \mathcal{G}} \quad \frac{\vdash \varepsilon : \mathcal{F}_1 \lhd \mathcal{G}}{\vdash l[\varepsilon] : \mathcal{F}_1 \oplus\!\!\!\!\!\!+\, \mathcal{F}_2 \lhd \mathcal{G}} \quad \frac{\vdash \varepsilon : \mathcal{F}_2 \lhd \mathcal{G}}{\vdash r[\varepsilon] : \mathcal{F}_1 \oplus\!\!\!\!\!\!+\, \mathcal{F}_2 \lhd \mathcal{G}}$$

Intuitively, $\vdash \varepsilon : \mathcal{F} \lhd \mathcal{G}$ (for a closed expression ε) means that ε denotes an element of $\mathcal{F}(\Omega_{\mathcal{G}})$, where $\Omega_{\mathcal{G}}$ is the final coalgebra of $\mathcal{G}$. As expected, there is a rule for each expression construct. The extra rule involving $\mathrm{Id} \lhd \mathcal{G}$ reflects the isomorphism between the final coalgebra $\Omega_{\mathcal{G}}$ and $\mathcal{G}(\Omega_{\mathcal{G}})$ (Lambek's lemma, cf. [35]). Only fixed-points at the outermost level of the functor are allowed. This does not mean however that we disallow nested fixed-points. For instance, $\mu x.\, a(x \oplus \mu y.\, a(y))$ would be a well-typed expression for the functor $\mathcal{D}$ of deterministic automata, as it will become clear below, when we will present more examples of well-typed and non-well-typed expressions. The presented type system is decidable (expressions are of finite length and the system is inductive on the structure of $\varepsilon \in \mathrm{Exp}$).

We can now formally define the set of $\mathcal{G}$-expressions: well-typed expressions associated with a non-deterministic functor $\mathcal{G}$.

Definition 2.6 ($\mathcal{G}$-*expressions*)**.** Let $\mathcal{G}$ be a non-deterministic functor and $\mathcal{F}$ an ingredient of $\mathcal{G}$. We define $\mathrm{Exp}_{\mathcal{F}\lhd\mathcal{G}}$ by:

$$\mathrm{Exp}_{\mathcal{F}\lhd\mathcal{G}} = \{\varepsilon \in \mathrm{Exp}^c \mid \vdash \varepsilon : \mathcal{F} \lhd \mathcal{G}\}\,.$$

We define the set $\mathrm{Exp}_{\mathcal{G}}$ of well-typed $\mathcal{G}$-*expressions* by $\mathrm{Exp}_{\mathcal{G}\lhd\mathcal{G}}$.

Examples of well-typed expressions for the functor $\mathcal{D} = 2 \times \mathrm{Id}^A$ (with $2 = \{0, 1\}$; recall that the ingredients of $\mathcal{D}$ are 2, Id^A and $\mathcal{D}$ itself) include $r\langle a(\underline{\emptyset})\rangle$, $l\langle 1\rangle \oplus r\langle a(l\langle 0\rangle)\rangle$ and $\mu x.r\langle a(x)\rangle \oplus l\langle 1\rangle$. The expressions $l[1]$, $l\langle 1\rangle \oplus 1$ and $\mu x.1$ are examples of non well-typed expressions, because the functor $\mathcal{D}$ does not involve $\oplus\!\!\!\!\!\!+$, the subexpressions in the sum have different types, and recursion is not at the outermost level (1 has type $2 \lhd \mathcal{D}$), respectively.

Let us instantiate the definition of $\mathcal{G}$-expressions to the functors of deterministic automata $\mathcal{D} = 2 \times \mathrm{Id}^A$.

Example 2.7 (*Deterministic expressions*)**.** Let A be a finite set of input actions and let X be a set of (recursion or) fixed-point variables. The set $\mathrm{Exp}_{\mathcal{D}}$ of *deterministic expressions* is given by the set of closed and guarded expressions generated by the following BNF grammar. For $a \in A$ and $x \in X$:

$$\begin{aligned}
\varepsilon &::= \underline{\emptyset} \mid \varepsilon \oplus \varepsilon \mid \mu x.\varepsilon \mid x \mid l\langle\varepsilon_1\rangle \mid r\langle\varepsilon_2\rangle\\
\varepsilon_1 &::= \underline{\emptyset} \mid 0 \mid 1 \mid \varepsilon_1 \oplus \varepsilon_1\\
\varepsilon_2 &::= \underline{\emptyset} \mid a(\varepsilon) \mid \varepsilon_2 \oplus \varepsilon_2
\end{aligned}$$

At this point, we should remark that the syntax of our expressions differs from the classical regular expressions in the use of μ and action prefixing $a(\varepsilon)$ instead of star and full concatenation. We shall show later that these two syntactically different formalisms are equally expressive (Theorem 2.12).

We have now defined a language of expressions which gives us an algebraic description of systems. The goal is now to present a generalization of Kleene's theorem for non-deterministic coalgebras (Theorem 2.12). Recall that, for regular languages, the theorem states that a language is regular if and only if it is recognized by a finite automaton. In order to achieve our goal we will first show that the set $\mathrm{Exp}_{\mathcal{G}}$ of $\mathcal{G}$-expressions carries a $\mathcal{G}$-coalgebra structure. More precisely, we are going to define a function

$$\delta_{\mathcal{F}\lhd\mathcal{G}} : \mathrm{Exp}_{\mathcal{F}\lhd\mathcal{G}} \to \mathcal{F}(\mathrm{Exp}_{\mathcal{G}})$$

for every ingredient $\mathcal{F}$ of $\mathcal{G}$, and then set $\delta_{\mathcal{G}} = \delta_{\mathcal{G}\lhd\mathcal{G}}$. Our definition of the function $\delta_{\mathcal{F}\lhd\mathcal{G}}$ will make use of the following.

Definition 2.8. For every $\mathcal{G} \in \mathit{NDF}$ and for every $\mathcal{F}$ with $\mathcal{F} \lhd \mathcal{G}$:

(i) we define a constant $\mathrm{Empty}_{\mathcal{F}\lhd\mathcal{G}} \in \mathcal{F}(\mathrm{Exp}_{\mathcal{G}})$ by induction on the syntactic structure of $\mathcal{F}$:

$$\begin{aligned}
\mathrm{Empty}_{\mathrm{Id}\lhd\mathcal{G}} &= \underline{\emptyset} & \mathrm{Empty}_{\mathcal{F}_1 \oplus\!\!\!\!\!\!+\, \mathcal{F}_2 \lhd \mathcal{G}} &= \bot\\
\mathrm{Empty}_{\mathrm{B}\lhd\mathcal{G}} &= \bot_{\mathrm{B}} & \mathrm{Empty}_{\mathcal{F}^A \lhd \mathcal{G}} &= \lambda a.\mathrm{Empty}_{\mathcal{F}\lhd\mathcal{G}}\\
\mathrm{Empty}_{\mathcal{F}_1\times\mathcal{F}_2\lhd\mathcal{G}} &= \langle \mathrm{Empty}_{\mathcal{F}_1\lhd\mathcal{G}}, \mathrm{Empty}_{\mathcal{F}_2\lhd\mathcal{G}}\rangle & \mathrm{Empty}_{\mathcal{P}_\omega\mathcal{F}\lhd\mathcal{G}} &= \{\}
\end{aligned}$$

(ii) we define a function $\mathrm{Plus}_{\mathcal{F}\lhd\mathcal{G}} : \mathcal{F}(\mathrm{Exp}_{\mathcal{G}}) \times \mathcal{F}(\mathrm{Exp}_{\mathcal{G}}) \to \mathcal{F}(\mathrm{Exp}_{\mathcal{G}})$ by induction on the syntactic structure of $\mathcal{F}$:

$$\begin{aligned}
\mathrm{Plus}_{\mathrm{Id}\lhd\mathcal{G}}(\varepsilon_1, \varepsilon_2) &= \varepsilon_1 \oplus \varepsilon_2\\
\mathrm{Plus}_{\mathrm{B}\lhd\mathcal{G}}(b_1, b_2) &= b_1 \vee_{\mathrm{B}} b_2
\end{aligned}$$

$$
\begin{aligned}
\mathrm{Plus}_{\mathcal{F}_1\times\mathcal{F}_2\triangleleft\mathcal{G}}(\langle\varepsilon_1,\varepsilon_2\rangle,\langle\varepsilon_3,\varepsilon_4\rangle) &= \langle \mathrm{Plus}_{\mathcal{F}_1\triangleleft\mathcal{G}}(\varepsilon_1,\varepsilon_3), \mathrm{Plus}_{\mathcal{F}_2\triangleleft\mathcal{G}}(\varepsilon_2,\varepsilon_4)\rangle\\
\mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{G}}(\kappa_i(\varepsilon_1),\kappa_i(\varepsilon_2)) &= \kappa_i(\mathrm{Plus}_{\mathcal{F}_i\triangleleft\mathcal{G}}(\varepsilon_1,\varepsilon_2)),\quad i\in\{1,2\}\\
\mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{G}}(\kappa_i(\varepsilon_1),\kappa_j(\varepsilon_2)) &= \top\quad i,j\in\{1,2\}\ \textit{and}\ i\neq j\\
\mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{G}}(x,\top) &= \mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{G}}(\top,x)=\top\\
\mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{G}}(x,\bot) &= \mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{G}}(\bot,x)=x\\
\mathrm{Plus}_{\mathcal{F}^A\triangleleft\mathcal{G}}(f,g) &= \lambda a.\,\mathrm{Plus}_{\mathcal{F}\triangleleft\mathcal{G}}(f(a),g(a))\\
\mathrm{Plus}_{\mathcal{P}_\omega\mathcal{F}\triangleleft\mathcal{G}}(s_1,s_2) &= s_1\cup s_2
\end{aligned}
$$

Intuitively, one can think of the constant $\mathrm{Empty}_{\mathcal{F}\triangleleft\mathcal{G}}$ and the function $\mathrm{Plus}_{\mathcal{F}\triangleleft\mathcal{G}}$ as liftings of $\underline{\emptyset}$ and $\oplus$ to the level of $\mathcal{F}(\mathrm{Exp}_{\mathcal{G}})$.

We need two more things to define $\delta_{\mathcal{F}\triangleleft\mathcal{G}}$. First, we define an order $\preceq$ on the types of expressions. For $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{G}$ non-deterministic functors such that $\mathcal{F}_1\triangleleft\mathcal{G}$ and $\mathcal{F}_2\triangleleft\mathcal{G}$, we define

$$(\mathcal{F}_1\triangleleft\mathcal{G})\preceq(\mathcal{F}_2\triangleleft\mathcal{G})\Leftrightarrow\mathcal{F}_1\triangleleft\mathcal{F}_2$$

The order $\preceq$ is a partial order (structure inherited from $\triangleleft$). Note also that $(\mathcal{F}_1\triangleleft\mathcal{G})=(\mathcal{F}_2\triangleleft\mathcal{G})\Leftrightarrow\mathcal{F}_1=\mathcal{F}_2$. Second, we define a measure $N(\varepsilon)$ based on the maximum number of nested unguarded occurrences of μ-expressions in ε and unguarded occurrences of $\oplus$. We say that a subexpression $\mu x.\varepsilon_1$ of ε occurs unguarded if it is not in the scope of one of the operators $l\langle-\rangle, r\langle-\rangle, l[-], r[-], a(-)$ or $\{-\}$.

Definition 2.9. For every guarded expression ε, we define $N(\varepsilon)$ as follows:

$$
\begin{aligned}
&N(\underline{\emptyset})=N(b)=N(a(\varepsilon))=N(l\langle\varepsilon\rangle)=N(r\langle\varepsilon\rangle)=N(l[\varepsilon])=N(r[\varepsilon])=N(\{\varepsilon\})=0\\
&N(\varepsilon_1\oplus\varepsilon_2)=1+max\{N(\varepsilon_1),\ N(\varepsilon_2)\}\\
&N(\mu x.\varepsilon)=1+N(\varepsilon)
\end{aligned}
$$

The measure N induces a partial order on the set of expressions: $\varepsilon_1\ll\varepsilon_2\Leftrightarrow N(\varepsilon_1)\leq N(\varepsilon_2)$, where $\leq$ is just the ordinary inequality of natural numbers.

Now we have all we need to define $\delta_{\mathcal{F}\triangleleft\mathcal{G}}:\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{G}}\to\mathcal{F}(\mathrm{Exp}_{\mathcal{G}})$.

Definition 2.10. For every ingredient $\mathcal{F}$ of a non-deterministic functor $\mathcal{G}$ and an expression $\varepsilon\in\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{G}}$, we define $\delta_{\mathcal{F}\triangleleft\mathcal{G}}(\varepsilon)$ as follows:

$$
\begin{aligned}
\delta_{\mathcal{F}\triangleleft\mathcal{G}}(\underline{\emptyset}) &= \mathrm{Empty}_{\mathcal{F}\triangleleft\mathcal{G}}\\
\delta_{\mathcal{F}\triangleleft\mathcal{G}}(\varepsilon_1\oplus\varepsilon_2) &= \mathrm{Plus}_{\mathcal{F}\triangleleft\mathcal{G}}(\delta_{\mathcal{F}\triangleleft\mathcal{G}}(\varepsilon_1),\delta_{\mathcal{F}\triangleleft\mathcal{G}}(\varepsilon_2))\\
\delta_{\mathcal{G}\triangleleft\mathcal{G}}(\mu x.\varepsilon) &= \delta_{\mathcal{G}\triangleleft\mathcal{G}}(\varepsilon[\mu x.\varepsilon/x])\\
\delta_{\mathrm{Id}\triangleleft\mathcal{G}}(\varepsilon) &= \varepsilon\quad\text{for } \mathcal{G}\neq\mathrm{Id}\\
\delta_{\mathrm{B}\triangleleft\mathcal{G}}(b) &= b\\
\delta_{\mathcal{F}_1\times\mathcal{F}_2\triangleleft\mathcal{G}}(l\langle\varepsilon\rangle) &= \langle\delta_{\mathcal{F}_1\triangleleft\mathcal{G}}(\varepsilon),\mathrm{Empty}_{\mathcal{F}_2\triangleleft\mathcal{G}}\rangle\\
\delta_{\mathcal{F}_1\times\mathcal{F}_2\triangleleft\mathcal{G}}(r\langle\varepsilon\rangle) &= \langle\mathrm{Empty}_{\mathcal{F}_1\triangleleft\mathcal{G}},\delta_{\mathcal{F}_2\triangleleft\mathcal{G}}(\varepsilon)\rangle\\
\delta_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{G}}(l[\varepsilon]) &= \kappa_1(\delta_{\mathcal{F}_1\triangleleft\mathcal{G}}(\varepsilon))\\
\delta_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{G}}(r[\varepsilon]) &= \kappa_2(\delta_{\mathcal{F}_2\triangleleft\mathcal{G}}(\varepsilon))\\
\delta_{\mathcal{F}^A\triangleleft\mathcal{G}}(a(\varepsilon)) &= \lambda a'.\begin{cases}\delta_{\mathcal{F}\triangleleft\mathcal{G}}(\varepsilon) & \text{if } a=a'\\ \mathrm{Empty}_{\mathcal{F}\triangleleft\mathcal{G}} & \text{otherwise}\end{cases}\\
\delta_{\mathcal{P}_\omega\mathcal{F}\triangleleft\mathcal{G}}(\{\varepsilon\}) &= \{\,\delta_{\mathcal{F}\triangleleft\mathcal{G}}(\varepsilon)\,\}
\end{aligned}
$$

Here, $\varepsilon[\mu x.\varepsilon/x]$ denotes syntactic substitution, replacing every free occurrence of x in ε by $\mu x.\varepsilon$.

In order to see that the definition of $\delta_{\mathcal{F}\triangleleft\mathcal{G}}$ is well-formed, we have to observe that $\delta_{\mathcal{F}\triangleleft\mathcal{G}}$ can be seen as a function having two arguments: the type $\mathcal{F}\triangleleft\mathcal{G}$ and the expression ε. Then, we use induction on the Cartesian product of types and expressions with orders $\preceq$ and $\ll$, respectively. More precisely, given two pairs $\langle\mathcal{F}_1\triangleleft\mathcal{G},\varepsilon_1\rangle$ and $\langle\mathcal{F}_2\triangleleft\mathcal{G},\varepsilon_2\rangle$ we have an order

$$
\begin{aligned}
\langle\mathcal{F}_1\triangleleft\mathcal{G},\varepsilon_1\rangle\leq\langle\mathcal{F}_2\triangleleft\mathcal{G},\varepsilon_2\rangle\ \Leftrightarrow\quad & \text{(i) } (\mathcal{F}_1\triangleleft\mathcal{G})\preceq(\mathcal{F}_2\triangleleft\mathcal{G})\\
& \text{or (ii) } (\mathcal{F}_1\triangleleft\mathcal{G})=(\mathcal{F}_2\triangleleft\mathcal{G}) \text{ and } \varepsilon_1\ll\varepsilon_2
\end{aligned}
\tag{2}
$$

Observe that in the definition above it is always true that $\langle \mathcal{F}' \triangleleft \mathcal{G}, \varepsilon' \rangle \leq \langle \mathcal{F} \triangleleft \mathcal{G}, \varepsilon \rangle$, for all occurrences of $\delta_{\mathcal{F}' \triangleleft \mathcal{G}}(\varepsilon')$ occurring in the right-hand side of the equation defining $\delta_{\mathcal{F} \triangleleft \mathcal{G}}(\varepsilon)$. In all cases, but the ones that ε is a fixed point or a sum expression, the inequality comes from point (i) above. For the case of the sum, note that $\langle \mathcal{F} \triangleleft \mathcal{G}, \varepsilon_1 \rangle \leq \langle \mathcal{F} \triangleleft \mathcal{G}, \varepsilon_1 \oplus \varepsilon_2 \rangle$ and $\langle \mathcal{F} \triangleleft \mathcal{G}, \varepsilon_2 \rangle \leq \langle \mathcal{F} \triangleleft \mathcal{G}, \varepsilon_1 \oplus \varepsilon_2 \rangle$ by point (ii), since $N(\varepsilon_1) < N(\varepsilon_1 \oplus \varepsilon_2)$ and $N(\varepsilon_2) < N(\varepsilon_1 \oplus \varepsilon_2)$. Similarly, in the case of $\mu x.\varepsilon$ we have that $N(\varepsilon) = N(\varepsilon[\mu x.\varepsilon/x])$, which can easily be proved by (standard) induction on the syntactic structure of ε, since ε is guarded (in x), and this guarantees that $N(\varepsilon[\mu x.\varepsilon/x]) < N(\mu x.\varepsilon)$. Hence, $\langle \mathcal{G} \triangleleft \mathcal{G}, \varepsilon \rangle \leq \langle \mathcal{G} \triangleleft \mathcal{G}, \mu x.\varepsilon \rangle$. Also note that clause 4 of the above definition overlaps with clauses 1 and 2 (by taking $\mathcal{F} = \mathrm{Id}$). However, they give the same result and thus the function $\delta_{\mathcal{F} \triangleleft \mathcal{G}}$ is well-defined.

Definition 2.11. We can now define, for each non-deterministic functor $\mathcal{G}$, a $\mathcal{G}$-coalgebra

$$\delta_{\mathcal{G}} : \mathrm{Exp}_{\mathcal{G}} \to \mathcal{G}(\mathrm{Exp}_{\mathcal{G}})$$

by putting $\delta_{\mathcal{G}} = \delta_{\mathcal{G} \triangleleft \mathcal{G}}$.

We remark that $\delta_{\mathcal{G}}$ is the generalization of the well-known notion of Brzozowski derivative [11] for regular expressions and, moreover, it provides an operational semantics for expressions. We now present the generalization of Kleene's theorem.

Theorem 2.12 ([8, Theorem 4])**.** *Let $\mathcal{G}$ be a non-deterministic functor.*

1. *For every locally finite $\mathcal{G}$-coalgebra (S, g) and for any $s \in S$ there exists an expression $\varepsilon_s \in \mathrm{Exp}_{\mathcal{G}}$ such that $\varepsilon_s \sim_b s$.*
2. *For every $\varepsilon \in \mathrm{Exp}_{\mathcal{G}}$, we can construct a coalgebra (S, g) such that S is finite and there exists $s_\varepsilon \in S$ with $\varepsilon \sim_b s_\varepsilon$.*

Note that $\varepsilon_s \sim_b s$ means that the expression ε_s and the (system with initial) state s have the same behaviour. For instance, for DFA's, this would mean that they denote and accept the same regular language. Similarly for ε and s_ε in item 2, above.

In [8], we presented a sound and complete axiomatization with respect to bisimilarity for $\mathrm{Exp}_{\mathcal{G}}$. We will not recall it here because this axiomatization can be recovered as an instance of the one presented in Section 4.

3. Monoidal exponentiation functor

In the previous section, we introduced non-deterministic functors and a language of expressions for specifying coalgebras. Coalgebras for non-deterministic functors cover many interesting types of systems, such as deterministic automata and Mealy machines, but not quantitative systems. For this reason, we recall the definition of the *monoidal exponentiation functor* [20], which will allow us to define coalgebras representing quantitative systems. In the next section, we will provide expressions and an axiomatization for these.

A *monoid* $\langle \mathbb{M}, +, 0 \rangle$ is an algebraic structure consisting of a set with an associative binary operation $+$ and a neutral element 0 for that operation. We will frequently refer to a monoid using the carrier set $\mathbb{M}$. A *commutative monoid* is a monoid where $+$ is also commutative. Examples of commutative monoids include 2, the two-element $\{0, 1\}$ Boolean algebra with logical "or", and the set $\mathbb{R}$ of real numbers with addition.

A property that will play a crucial role in the rest of the paper is *idempotency*: a monoid is idempotent, if the associated binary operation $+$ is idempotent. For example, the monoid 2 is idempotent, whilst $\mathbb{R}$ is not. Note that an idempotent commutative monoid is a join-semilattice.

For a function φ from a set S to a monoid $\mathbb{M}$, we define *support of* φ as the set $\{s \in S \mid \varphi(s) \neq 0\}$.

Definition 3.1 (*Monoidal exponentiation functor*)**.** Let $\langle \mathbb{M}, +, 0 \rangle$ be a commutative monoid. The monoidal exponentiation functor $\mathbb{M}^-_\omega$: **Set** $\to$ **Set** is defined as follows. For each set S, $\mathbb{M}^S_\omega$ is the set of functions from S to $\mathbb{M}$ with finite support. For each function $h\colon S \to T$, $\mathbb{M}^h_\omega\colon \mathbb{M}^S_\omega \to \mathbb{M}^T_\omega$ is the function mapping each $\varphi \in \mathbb{M}^S_\omega$ into $\varphi^h \in \mathbb{M}^T_\omega$ defined, for every $t \in T$, as

$$\varphi^h(t) = \sum_{s' \in h^{-1}(t)} \varphi(s')$$

Throughout this paper we will omit the subscript ω and use $\mathbb{M}^-$ to denote the monoidal exponentiation functor. Note that the (finite) powerset functor $\mathcal{P}_\omega(-)$ coincides with 2^-_ω. This is often used to represent non-deterministic systems. For example, LTS's (with labels over A) are coalgebras of the functor $\mathcal{P}_\omega(-)^A$.

Proposition 3.2. *The functor $\mathbb{M}^-$ is bounded.*

Proof. Using [22, Theorem 4.7] it is enough to prove that there exists a natural surjection η from a functor $B \times (-)^A$ to $\mathbb{M}^-$, for some sets B and A.

We take $A = \mathbb{N}$ and $B = \mathbb{M}^{\mathbb{N}}$, where $\mathbb{N}$ denotes the set of all natural numbers and we define for every set X the function $\eta_X : \mathbb{M}^{\mathbb{N}} \times X^{\mathbb{N}} \to \mathbb{M}^X$ as

$$\eta_X(\varphi, f)(x) = \sum_{n \in f^{-1}(x)} \varphi(n)$$

The function η_X is surjective. It remains to prove that it is natural. Take $h : X \to Y$. We shall prove that the following diagram commutes

$$\begin{array}{ccc} \mathbb{M}^{\mathbb{N}} \times X^{\mathbb{N}} & \xrightarrow{id \times h^{\mathbb{N}}} & \mathbb{M}^{\mathbb{N}} \times Y^{\mathbb{N}} \\ \downarrow{\scriptstyle \eta_X} & & \downarrow{\scriptstyle \eta_Y} \\ \mathbb{M}^X & \xrightarrow[\mathbb{M}^h]{} & \mathbb{M}^Y \end{array}$$

that is $\mathbb{M}^h \circ \eta_X = \eta_Y \circ (id \times h^{\mathbb{N}})$.

$$\begin{aligned} (\mathbb{M}^h \circ \eta_X)(\varphi, f) &= \textstyle\sum_{x \in h^{-1}(y)} \eta_X(\varphi, f)(x) && \text{(def. } \mathbb{M}^h \text{ applied to } \eta_X(\varphi, f)) \\ &= \textstyle\sum_{x \in h^{-1}(y)} \sum_{n \in f^{-1}(x)} \varphi(n) && \text{(def. } \eta_X) \\ &= \textstyle\sum_{n \in (h \circ f)^{-1}(y)} \varphi(n) && (f \text{ and } h \text{ are functions}) \\ &= \eta_Y(\varphi, h \circ f) && \text{(def. } \eta_Y) \\ &= (\eta_Y \circ (id \times h^{\mathbb{N}}))(\varphi, f) && \square \end{aligned}$$

Corollary 3.3. *The functor $\mathbb{M}^-$ has a final coalgebra.*

Proof. By [21, Theorem 7.2], the fact that $\mathbb{M}^-$ is bounded is enough to guarantee the existence of a final coalgebra. □

Recall that $\mathbb{M}^-$ does not preserve weak-pullbacks [20], but it preserves arbitrary intersections [20, Corollary 5.4], which we need to define smallest subcoalgebras.

We finish this section with an example of quantitative systems – weighted automata – modelled as coalgebras of a functor which contains the monoidal exponentiation as a subfunctor.

Weighted automata. Weighted automata [18,38] are transition systems labelled over a set A and with weights in a semiring $\mathbb{S}$. Moreover, each state is equipped with an output value [1] in $\mathbb{S}$. A *semiring* $\mathbb{S}$ is a tuple $\langle \mathbb{S}, +, \times, 0, 1 \rangle$ where $\langle \mathbb{S}, +, 0 \rangle$ is a commutative monoid and $\langle \mathbb{S}, \times, 1 \rangle$ is a monoid satisfying certain distributive laws. Examples of semirings include the real numbers $\mathbb{R}$, with usual addition and multiplication, and the Boolean semiring 2 with disjunction and conjunction.

From a coalgebraic perspective, weighted automata are coalgebras of the functor $\mathcal{W} = \mathbb{S} \times (\mathbb{S}^{\mathrm{Id}})^A$, where we write again $\mathbb{S}$ to denote the commutative monoid of the semiring $\mathbb{S}$. More concretely, a coalgebra for $\mathbb{S} \times (\mathbb{S}^{\mathrm{Id}})^A$ is a pair $(S, \langle o, T \rangle)$, where S is a set of states, $o : S \to \mathbb{S}$ is the function that associates an output weight to each state $s \in S$ and $T : S \to (\mathbb{S}^S)^A$ is the transition relation that associates a weight to each transition. We will use the following notation in the representation of weighted automata:

$$\begin{array}{ccc} s & \xrightarrow{a,w} & s' \\ \downarrow & & \downarrow \\ o_s & & o_{s'} \end{array} \Leftrightarrow T(s)(a)(s') = w \text{ and } o(s) = o_s \text{ and } o(s') = o_{s'}$$

If the set of states S and the alphabet A are finite, weighted automata can be conveniently represented in the following way. Let $S = \{s_1, \ldots, s_n\}$ be the set of states and $A = \{a_1, \ldots, a_m\}$ the input alphabet. The output function o can be seen as a vector with n entries

$$o = \begin{pmatrix} o(s_1) \\ \vdots \\ o(s_n) \end{pmatrix}$$

[1] In the original formulation also an input value is considered. To simplify the presentation and following [13] we omit it.

and the transition function T is a set of m matrices (of dimension $n \times n$)

$$T_{a_i} = \begin{pmatrix} t_{11} & \dots & t_{1n} \\ \vdots & & \vdots \\ t_{n1} & \dots & t_{nn} \end{pmatrix} \text{ with } t_{jk} = T(s_j)(a_i)(s_k)$$

This representation has advantages in the definition of homomorphism between two weighted automata. Composition of homomorphisms can be expressed as matrix multiplication, making it easier to check the commutativity of the diagram below. This will be useful in the proof of Proposition 3.4, which states the coincidence between the coalgebraic notion of behavioural equivalence for the weighted automata functor and the bisimilarity notion introduced in [12].

Let $(S, \langle o, T\rangle)$ and $(S', \langle o', T'\rangle)$ be two weighted automata. A homomorphism between these automata is a function $h\colon S \to S'$ which makes the following diagram commute

$$\begin{array}{ccc} S & \xrightarrow{\quad h \quad} & S' \\ {\scriptstyle \langle o,T\rangle}\downarrow & & \downarrow{\scriptstyle \langle o',T'\rangle} \\ \mathbb{S} \times (\mathbb{S}^S)^A & \xrightarrow[id\times(\mathbb{S}^h)^A]{} & \mathbb{S} \times (\mathbb{S}^{S'})^A \end{array}$$

Now, representing $h\colon S \to S'$, with $S = \{s_1, \dots, s_n\}$ and $S' = \{s'_1, \dots, s'_m\}$ as a matrix with dimensions $n \times m$ in the following way

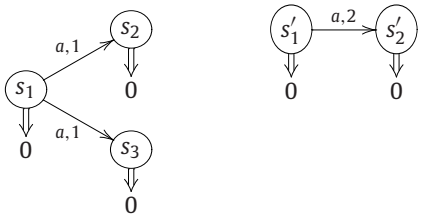
$$h = \begin{pmatrix} h_{11} & \dots & h_{1m} \\ \vdots & & \vdots \\ h_{n1} & \dots & h_{nm} \end{pmatrix} \text{ with } h_{jk} = \begin{cases} 1 & h(s_j) = s'_k \\ 0 & \text{otherwise} \end{cases}$$

one can formulate the commutativity condition of the diagram above – $(id \times (\mathbb{S}^h)^A) \circ \langle o, T\rangle = \langle o', T'\rangle \circ h$ – as the following matrix equalities:

$$o = h \times o' \text{ and } \forall_{a\in A}\ T_a \times h = h \times T'_a$$

Here, we are using the same letters to denote the functions, on the left side of the equations, and their representation as matrices, on the right side. Note that $(\mathbb{S}^h \circ T_a)(s_i)(s'_j) = (T_a \times h)(i, j)$, $T'_a \circ h = h \times T'_a$ and $o' \circ h = h \times o'$.

For a concrete example, let $\mathbb{S} = \mathbb{R}$, let $A = \{a\}$ and consider the two weighted automata depicted below.



$$o = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad T_a = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad o' = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad T'_a = \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix}$$

Now consider the morphism $h\colon S \to S'$ which maps s_1 to s'_1 and both s_2 and s_3 to s'_2, that is, it corresponds to the matrix

$$h = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

We now compute

$$h \times o' = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = o \text{ and } T_a \times h = \begin{pmatrix} 0 & 2 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} = h \times T'_a$$

and we can conclude that h is a coalgebra homomorphism.

It is worth recalling that coalgebra homomorphisms always map states into bisimilar states ([35, Lemma 5.3]). Thus, since h is a $\mathbb{R} \times (\mathbb{R}^{\mathrm{Id}})^A$-homomorphism, s_1 is bisimilar to s'_1 and s_2,s_3 are bisimilar to s'_2.

Note that the multiplicative monoid $\langle \mathbb{S}, \times, 1 \rangle$ plays no role in the coalgebraic definition of weighted automata. Also in [18,38] it is used only to define the weight of a sequence of transitions. Bisimilarity for weighted automata has been studied in [12] and it coincides with the coalgebraic notion of behavioural equivalence.

Proposition 3.4. *Behavioural equivalence for $\mathbb{S} \times (\mathbb{S}^{\mathrm{Id}})^A$ coincides with the weighted automata bisimilarity defined in [12].*

Proof. The definition of homomorphism which we stated above using matrix multiplication coincides with the definition of functional simulation [12, Definition 3.1]. Then, by [12, Corollary 3.6], states $s \in S$ and $s' \in S'$ of two weighted automata $(S, \langle o, T \rangle)$ and $(S', \langle o', T' \rangle)$ are bisimilar (according to [12]) if and only if there exists a weighted automata $\langle Q, \langle o_1, T_1 \rangle)$ such that there exist surjective functional simulations $h\colon S \to Q$ and $h'\colon S' \to Q$ satisfying $h(s) = h'(s')$. In coalgebraic terms, h and h' form a cospan of coalgebra homomorphisms, which we show in the following commuting diagram:

$$\begin{array}{ccccc} S & \xrightarrow{\ h\ } & Q & \xleftarrow{\ h'\ } & S' \\ {\scriptstyle \langle o,T \rangle}\downarrow & & \downarrow{\scriptstyle \langle o_1,T_1 \rangle} & & \downarrow{\scriptstyle \langle o',T' \rangle} \\ \mathbb{S} \times (\mathbb{S}^S)^A & \longrightarrow & \mathbb{S} \times (\mathbb{S}^Q)^A & \longleftarrow & \mathbb{S} \times (\mathbb{S}^{S'})^A \end{array}$$

Thus, for any $s \in S$ and $s' \in S'$, if they are bisimilar accordding to [12], that is if $h(s) = h'(s')$, then we have that $\mathbf{beh}_Q(h(s)) = \mathbf{beh}_Q(h'(s'))$ which, by uniqueness of the map into the final coalgebra, implies that $\mathbf{beh}_S(s) = \mathbf{beh}_{S'}(s')$. Hence, the states s and s' are behaviourally equivalent.

For the converse implication, suppose that s and s' are behaviourally equivalent, that is $\mathbf{beh}_S(s) = \mathbf{beh}_{S'}(s')$. We set $(S, \langle o, T \rangle)$, $(S', \langle o', T' \rangle)$ and $(Q, \langle o_1, T_1 \rangle)$ in the diagram above to be the subcoalgebras $\langle s \rangle$, $\langle s' \rangle$ and $\mathbf{beh}_S(\langle s \rangle)$. The key point is now to observe that $\mathbf{beh}_S(\langle s \rangle) = \mathbf{beh}_{S'}(\langle s' \rangle)$ and thus, by definition, we have two surjective homomorphisms h and h' (the suitable restrictions of $\mathbf{beh}_S$ and $\mathbf{beh}_{S'}$ to $\langle s \rangle$ and $\langle s' \rangle$, respectively) satisfying $h(s) = h'(s')$. Hence, s and s' are bisimilar according to [12]. □

4. A non-idempotent algebra for quantitative regular behaviours

In this section, we will extend the framework presented in Section 2 in order to deal with quantitative systems, as described in the previous section. We will start by defining an appropriate class of functors $\mathcal{H}$, proceed with presenting the language $\mathrm{Exp}_{\mathcal{H}}$ of expressions associated with $\mathcal{H}$ together with a Kleene like theorem and finally we introduce an axiomatization of $\mathrm{Exp}_{\mathcal{H}}$ and prove it sound and complete with respect to behavioural equivalence.

Definition 4.1. The set *QF* of *quantitative functors* on **Set** is defined inductively by putting:

$$QF \ni \mathcal{H} ::= \mathcal{G} \mid \mathbb{M}^{\mathcal{H}} \mid (\mathbb{M}^{\mathcal{H}})^A \mid \mathbb{M}_1^{\mathcal{H}_1} \times \mathbb{M}_2^{\mathcal{H}_2} \mid \mathbb{M}_1^{\mathcal{H}_1} \oplus \mathbb{M}_2^{\mathcal{H}_2}$$

where $\mathcal{G}$ is a non-deterministic functor, $\mathbb{M}$ is a commutative monoid and A is a finite set.

Note that the $\mathcal{P}_\omega$ functor is explicitly included in the syntax above, since it is a non-deterministic functor. Moreover, note that we do not allow *mixed* functors, such as $\mathcal{G} \oplus \mathbb{M}^{\mathcal{H}}$ or $\mathcal{G} \times \mathbb{M}^{\mathcal{H}}$. The reason for this restriction will become clear later in this section when we discuss the proof of Kleene's theorem. In Section 5, we will show how to deal with such mixed functors.

We need now to extend several definitions presented in Section 2. The definition of the ingredient associated with a functor is extended in the expected way, as we show next.

Definition 4.2. Let $\lhd \subseteq QF \times QF$ be the least reflexive and transitive relation on quantitative functors such that

$$\mathcal{H}_1 \lhd \mathcal{H}_1 \times \mathcal{H}_2,\ \mathcal{H}_2 \lhd \mathcal{H}_1 \times \mathcal{H}_2,\ \mathcal{H}_1 \lhd \mathcal{H}_1 \oplus \mathcal{H}_2,\ \mathcal{H}_2 \lhd \mathcal{H}_1 \oplus \mathcal{H}_2,\ \mathcal{H} \lhd \mathcal{H}^A,\ \mathcal{H} \lhd \mathcal{P}_\omega \mathcal{H},\ \mathcal{H} \lhd \mathbb{M}^{\mathcal{H}}$$

All the other definitions we presented in Section 2 need now to be extended to quantitative functors. We start by observing that taking the current definitions and replacing the subscript $\mathcal{F} \lhd \mathcal{G}$ with $\mathcal{F} \lhd \mathcal{H}$ does most of the work. In fact, having that, we just need to extend all the definitions for the case $\mathbb{M}^{\mathcal{F}} \lhd \mathcal{H}$.

We start by introducing a new expression $m \cdot \varepsilon$, which we highlight in the definition, with $m \in \mathbb{M}$, extending the set of untyped expressions.

Definition 4.3 (*Expressions for quantitative functors*)**.** Let A be a finite set, B a finite join-semilattice, $\mathbb{M}$ a commutative monoid and X a set of fixed-point variables. The set of all *expressions* is given by the following grammar, where $a \in A$, $b \in \mathrm{B}$, $m \in \mathbb{M}$ and $x \in X$:

$$\varepsilon ::= \underline{\emptyset} \mid x \mid \varepsilon \oplus \varepsilon \mid \mu x.\gamma \mid b \mid l\langle\varepsilon\rangle \mid r\langle\varepsilon\rangle \mid l[\varepsilon] \mid r[\varepsilon] \mid a(\varepsilon) \mid \{\varepsilon\} \mid m \cdot \varepsilon$$

where γ is a *guarded expression* given by:

$$\gamma ::= \underline{\emptyset} \mid \gamma \oplus \gamma \mid \mu x.\gamma \mid b \mid l\langle\varepsilon\rangle \mid r\langle\varepsilon\rangle \mid l[\varepsilon] \mid r[\varepsilon] \mid a(\varepsilon) \mid \{\varepsilon\} \mid m \cdot \varepsilon$$

The intuition behind the new expression $m \cdot \varepsilon$ is that there is a transition between the current state and the state specified by ε with weight m.

The type system will have one extra rule, which we highlight in the definition.

Definition 4.4 (*Type system*)**.** We now define a typing relation $\vdash \subseteq \mathrm{Exp} \times QF \times QF$ that will associate an expression ε with two quantitative functors $\mathcal{F}$ and $\mathcal{H}$, which are related by the ingredient relation ($\mathcal{F}$ is an ingredient of $\mathcal{H}$). We shall write $\vdash \varepsilon : \mathcal{F} \lhd \mathcal{H}$ for $\langle \varepsilon, \mathcal{F}, \mathcal{H}\rangle \in \vdash$. The rules that define $\vdash$ are the following:

$$\frac{}{\vdash \underline{\emptyset} : \mathcal{F} \lhd \mathcal{H}} \qquad \frac{}{\vdash b : \mathrm{B} \lhd \mathcal{H}} \qquad \frac{}{\vdash x : \mathcal{H} \lhd \mathcal{H}} \qquad \frac{\vdash \varepsilon : \mathcal{H} \lhd \mathcal{H}}{\vdash \mu x.\varepsilon : \mathcal{H} \lhd \mathcal{H}}$$

$$\frac{\vdash \varepsilon_1 : \mathcal{F} \lhd \mathcal{H} \quad \vdash \varepsilon_2 : \mathcal{F} \lhd H}{\vdash \varepsilon_1 \oplus \varepsilon_2 : \mathcal{F} \lhd \mathcal{H}} \qquad \frac{\vdash \varepsilon : \mathcal{H} \lhd \mathcal{H}}{\vdash \varepsilon : \mathrm{Id} \lhd \mathcal{H}} \qquad \frac{\vdash \varepsilon : \mathcal{F} \lhd \mathcal{H}}{\vdash \{\varepsilon\} : \mathcal{P}_\omega \mathcal{F} \lhd \mathcal{H}} \qquad \frac{\vdash \varepsilon : \mathcal{F} \lhd \mathcal{H}}{\vdash a(\varepsilon) : \mathcal{F}^A \lhd \mathcal{H}}$$

$$\frac{\vdash \varepsilon : \mathcal{F}_1 \lhd \mathcal{H}}{\vdash l\langle\varepsilon\rangle : \mathcal{F}_1 \times \mathcal{F}_2 \lhd \mathcal{H}} \qquad \frac{\vdash \varepsilon : \mathcal{F}_2 \lhd \mathcal{H}}{\vdash r\langle\varepsilon\rangle : \mathcal{F}_1 \times \mathcal{F}_2 \lhd \mathcal{H}} \qquad \frac{\vdash \varepsilon : \mathcal{F}_1 \lhd \mathcal{H}}{\vdash l[\varepsilon] : \mathcal{F}_1 \oplus \mathcal{F}_2 \lhd \mathcal{H}} \qquad \frac{\vdash \varepsilon : \mathcal{F}_2 \lhd \mathcal{H}}{\vdash r[\varepsilon] : \mathcal{F}_1 \oplus \mathcal{F}_2 \lhd \mathcal{H}}$$

$$\frac{\vdash \varepsilon : \mathcal{F} \lhd \mathcal{H}}{\vdash m \cdot \varepsilon : \mathbb{M}^{\mathcal{F}} \lhd \mathcal{H}}$$

We define $\mathrm{Exp}_{\mathcal{F} \lhd \mathcal{H}}$ by:

$$\mathrm{Exp}_{\mathcal{F} \lhd \mathcal{H}} = \{\varepsilon \in \mathrm{Exp} \mid \vdash \varepsilon : \mathcal{F} \lhd \mathcal{H}\}.$$

The set $\mathrm{Exp}_{\mathcal{H}}$ of well-typed $\mathcal{H}$-*expressions* equals $\mathrm{Exp}_{\mathcal{H} \lhd \mathcal{H}}$.

Next, we provide the set $\mathrm{Exp}_{\mathcal{H}}$ with a coalgebraic structure. More precisely, we define a function $\delta_{\mathcal{F} \lhd \mathcal{H}} : \mathrm{Exp}_{\mathcal{F} \lhd \mathcal{H}} \to \mathcal{F}(\mathrm{Exp}_{\mathcal{H}})$ and then set $\delta_{\mathcal{H}} = \delta_{\mathcal{H} \lhd \mathcal{H}}$. We show the definition of $\delta_{\mathcal{F} \lhd \mathcal{H}}$ as well as of the auxiliary constant $\mathrm{Empty}_{\mathcal{F} \lhd \mathcal{H}}$ and function $\mathrm{Plus}_{\mathcal{F} \lhd H}$. As before we highlight the new part of the definition when compared with the definition for non-deterministic functors.

Definition 4.5. For every $\mathcal{H} \in QF$ and for every $\mathcal{F}$ with $\mathcal{F} \lhd \mathcal{H}$:

(i) we define a constant $\mathrm{Empty}_{\mathcal{F} \lhd \mathcal{H}} \in \mathcal{F}(\mathrm{Exp}_{\mathcal{H}})$ by induction on the syntactic structure of $\mathcal{F}$:

$$\begin{aligned}
\mathrm{Empty}_{\mathrm{Id} \lhd \mathcal{H}} &= \underline{\emptyset} & \mathrm{Empty}_{\mathcal{F}_1 \oplus \mathcal{F}_2 \lhd \mathcal{H}} &= \bot \\
\mathrm{Empty}_{\mathrm{B} \lhd \mathcal{H}} &= \bot_{\mathrm{B}} & \mathrm{Empty}_{\mathcal{F}^A \lhd \mathcal{H}} &= \lambda a.\mathrm{Empty}_{\mathcal{F} \lhd \mathcal{H}} \\
\mathrm{Empty}_{\mathcal{F}_1 \times \mathcal{F}_2 \lhd \mathcal{H}} &= \langle \mathrm{Empty}_{\mathcal{F}_1 \lhd \mathcal{H}}, \mathrm{Empty}_{\mathcal{F}_2 \lhd \mathcal{H}} \rangle & \mathrm{Empty}_{\mathcal{P}_\omega \mathcal{F} \lhd \mathcal{H}} &= \{\} \\
\mathrm{Empty}_{\mathbb{M}^{\mathcal{F}} \lhd \mathcal{H}} &= \lambda c.0
\end{aligned}$$

(ii) we define a function $\mathrm{Plus}_{\mathcal{F} \lhd \mathcal{H}} : \mathcal{F}(\mathrm{Exp}_{\mathcal{H}}) \times \mathcal{F}(\mathrm{Exp}_{\mathcal{H}}) \to \mathcal{F}(\mathrm{Exp}_{\mathcal{H}})$ by induction on the syntactic structure of $\mathcal{F}$:

$$\begin{aligned}
\mathrm{Plus}_{\mathrm{Id} \lhd \mathcal{H}}(\varepsilon_1, \varepsilon_2) &= \varepsilon_1 \oplus \varepsilon_2 \\
\mathrm{Plus}_{\mathrm{B} \lhd \mathcal{H}}(b_1, b_2) &= b_1 \vee_{\mathrm{B}} b_2
\end{aligned}$$

$$
\begin{aligned}
\mathrm{Plus}_{\mathcal{F}_1\times\mathcal{F}_2\triangleleft\mathcal{H}}(\langle\varepsilon_1,\varepsilon_2\rangle,\langle\varepsilon_3,\varepsilon_4\rangle) &= \langle \mathrm{Plus}_{\mathcal{F}_1\triangleleft\mathcal{H}}(\varepsilon_1,\varepsilon_3), \mathrm{Plus}_{\mathcal{F}_2\triangleleft\mathcal{H}}(\varepsilon_2,\varepsilon_4)\rangle\\
\mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{H}}(\kappa_i(\varepsilon_1),\kappa_i(\varepsilon_2)) &= \kappa_i(\mathrm{Plus}_{\mathcal{F}_i\triangleleft\mathcal{H}}(\varepsilon_1,\varepsilon_2)),\quad i\in\{1,2\}\\
\mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{H}}(\kappa_i(\varepsilon_1),\kappa_j(\varepsilon_2)) &= \top\ \ i,j\in\{1,2\}\ \textit{and}\ i\neq j\\
\mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{H}}(x,\top) &= \mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{H}}(\top,x)=\top\\
\mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{H}}(x,\bot) &= \mathrm{Plus}_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{H}}(\bot,x)=x\\
\mathrm{Plus}_{\mathcal{F}^A\triangleleft\mathcal{H}}(f,g) &= \lambda a.\ \mathrm{Plus}_{\mathcal{F}\triangleleft H}(f(a),g(a))\\
\mathrm{Plus}_{\mathcal{P}_\omega\mathcal{F}\triangleleft\mathcal{H}}(s_1,s_2) &= s_1\cup s_2\\
\mathrm{Plus}_{\mathbb{M}^{\mathcal{F}}\triangleleft\mathcal{H}}(f,g) &= \lambda c.f(c)+g(c)
\end{aligned}
$$

(iii) we define a function $\delta_{\mathcal{F}\triangleleft\mathcal{H}}:\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}\to\mathcal{F}(\mathrm{Exp}_{\mathcal{H}})$, by induction on the product of types of expressions and expressions (using the order defined in Eq. (2), extended with the clause $N(m\cdot\varepsilon)=0$). For every ingredient $\mathcal{F}$ of a non-deterministic functor $\mathcal{H}$ and an expression $\varepsilon\in\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}$, we define $\delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon)$ as follows:

$$
\begin{aligned}
\delta_{\mathcal{F}\triangleleft\mathcal{H}}(\underline{\emptyset}) &= \mathrm{Empty}_{\mathcal{F}\triangleleft\mathcal{H}}\\
\delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon_1\oplus\varepsilon_2) &= \mathrm{Plus}_{\mathcal{F}\triangleleft\mathcal{H}}(\delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon_1),\delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon_2))\\
\delta_{\mathcal{H}\triangleleft\mathcal{H}}(\mu x.\varepsilon) &= \delta_{\mathcal{H}\triangleleft\mathcal{H}}(\varepsilon[\mu x.\varepsilon/x])\\
\delta_{\mathrm{Id}\triangleleft\mathcal{H}}(\varepsilon) &= \varepsilon\quad \text{for } \mathcal{H}\neq\mathrm{Id}\\
\delta_{\mathrm{B}\triangleleft\mathcal{H}}(b) &= b\\
\delta_{\mathcal{F}_1\times\mathcal{F}_2\triangleleft\mathcal{H}}(l\langle\varepsilon\rangle) &= \langle\delta_{\mathcal{F}_1\triangleleft\mathcal{H}}(\varepsilon),\mathrm{Empty}_{\mathcal{F}_2\triangleleft\mathcal{H}}\rangle\\
\delta_{\mathcal{F}_1\times\mathcal{F}_2\triangleleft\mathcal{H}}(r\langle\varepsilon\rangle) &= \langle\mathrm{Empty}_{\mathcal{F}_1\triangleleft\mathcal{H}},\delta_{\mathcal{F}_2\triangleleft\mathcal{H}}(\varepsilon)\rangle\\
\delta_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{H}}(l[\varepsilon]) &= \kappa_1(\delta_{\mathcal{F}_1\triangleleft\mathcal{H}}(\varepsilon))\\
\delta_{\mathcal{F}_1\oplus\mathcal{F}_2\triangleleft\mathcal{H}}(r[\varepsilon]) &= \kappa_2(\delta_{\mathcal{F}_2\triangleleft\mathcal{H}}(\varepsilon))\\
\delta_{\mathcal{F}^A\triangleleft\mathcal{H}}(a(\varepsilon)) &= \lambda a'.\begin{cases}\delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon) & \text{if } a=a'\\ \mathrm{Empty}_{\mathcal{F}\triangleleft\mathcal{H}} & \text{otherwise}\end{cases}\\
\delta_{\mathcal{P}_\omega\mathcal{F}\triangleleft\mathcal{G}}(\{\varepsilon\}) &= \{\,\delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon)\,\}\\
\delta_{\mathbb{M}^{\mathcal{F}}\triangleleft\mathcal{H}}(m\cdot\varepsilon) &= \lambda c.\begin{cases}m & \text{if } \delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon)=c\\ 0 & \text{otherwise}\end{cases}
\end{aligned}
$$

The function $\delta_{\mathcal{H}}=\delta_{\mathcal{H}\triangleleft\mathcal{H}}$ provides an operational semantics for the expressions. We will soon illustrate this for the case of expressions for weighted automata (Example 4.8).

Finally, we introduce an equational system for expressions of type $\mathcal{F}\triangleleft H$. We define the relation $\equiv\subseteq\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}\times\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}$, written infix, as the least reflexive and transitive relation containing the following identities:

$$
\begin{array}{llll}
\varepsilon\oplus\varepsilon\equiv\varepsilon, & \text{if } \varepsilon\in\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{G}} & \textit{(Idempotency)} & \\
\varepsilon_1\oplus\varepsilon_2\equiv\varepsilon_2\oplus\varepsilon_1 & & \textit{(Commutativity)} & \gamma[\mu x.\gamma/x]\equiv\mu x.\gamma \quad \textit{(FP)}\\
\varepsilon_1\oplus(\varepsilon_2\oplus\varepsilon_3)\equiv(\varepsilon_1\oplus\varepsilon_2)\oplus\varepsilon_3 & & \textit{(Associativity)} & \gamma[\varepsilon/x]\equiv\varepsilon\Rightarrow\mu x.\gamma\equiv\varepsilon \quad \textit{(Unique)}\\
\underline{\emptyset}\oplus\varepsilon\equiv\varepsilon & & \textit{(Empty)} &
\end{array}
$$

$$
\begin{array}{llll}
\underline{\emptyset}\equiv\bot_{\mathrm{B}} & (\mathrm{B}-\underline{\emptyset}) & b_1\oplus b_2\equiv b_1\vee_{\mathrm{B}} b_2 & (\mathrm{B}-\oplus)\\
l\langle\underline{\emptyset}\rangle\equiv\underline{\emptyset} & (\times-\underline{\emptyset}-L) & l\langle\varepsilon_1\oplus\varepsilon_2\rangle\equiv l\langle\varepsilon_1\rangle\oplus l\langle\varepsilon_2\rangle & (\times-\oplus-L)\\
r\langle\underline{\emptyset}\rangle\equiv\underline{\emptyset} & (\times-\underline{\emptyset}-R) & r\langle\varepsilon_1\oplus\varepsilon_2\rangle\equiv r\langle\varepsilon_1\rangle\oplus r\langle\varepsilon_2\rangle & (\times-\oplus-R)\\
a(\underline{\emptyset})\equiv\underline{\emptyset} & (-^A-\underline{\emptyset}) & a(\varepsilon_1\oplus\varepsilon_2)\equiv a(\varepsilon_1)\oplus a(\varepsilon_2) & (-^A-\oplus)\\
(0\cdot\varepsilon)\equiv\underline{\emptyset} & (\mathbb{M}^-\underline{\emptyset}) & (m\cdot\varepsilon)\oplus(m'\cdot\varepsilon)\equiv(m+m')\cdot\varepsilon & (\mathbb{M}^-\oplus)\\
 & & l[\varepsilon_1\oplus\varepsilon_2]\equiv l[\varepsilon_1]\oplus l[\varepsilon_2] & (+-\oplus-L)\\
 & & r[\varepsilon_1\oplus\varepsilon_2]\equiv r[\varepsilon_1]\oplus r[\varepsilon_2] & (+-\oplus-R)\\
 & & l[\varepsilon_1]\oplus r[\varepsilon_2]\equiv l[\underline{\emptyset}]\oplus r[\underline{\emptyset}] & (+-\oplus-\top)
\end{array}
$$

$$\varepsilon_1 \equiv \varepsilon_2 \Rightarrow \varepsilon[\varepsilon_1/x] \equiv \varepsilon[\varepsilon_2/x] \quad \text{if } x \text{ is free in } \varepsilon \qquad (Cong)$$

$$\mu x.\gamma \equiv \mu y.\gamma[y/x] \quad \text{if } y \text{ is not free in } \gamma \qquad (\alpha - equiv)$$

Note that (*Idempotency*) only holds for $\varepsilon \in \mathrm{Exp}_{\mathcal{F}\lhd\mathcal{G}}$. The reason why it cannot hold for the remaining functors comes from the fact that a monoid is, in general, not idempotent. Thus, (*Idempotency*) would conflict with the axiom $(\mathbb{M}^- - \oplus)$, which allows us to derive, for instance, $(2 \cdot \underline{\emptyset}) \oplus (2 \cdot \underline{\emptyset}) \equiv 4 \cdot \underline{\emptyset}$. In the case of an idempotent commutative monoid $\mathbb{M}$, (*Idempotency*) follows from the axiom $(\mathbb{M}^- - \oplus)$.

Lemma 4.6. *Let $\mathbb{M}$ be an idempotent commutative monoid. For every expression $\varepsilon \in \mathrm{Exp}_{\mathbb{M}^{\mathcal{F}}\lhd\mathcal{H}}$, one has $\varepsilon \oplus \varepsilon \equiv \varepsilon$.*

Proof. By induction on the product of types of expressions and expressions (using the order defined in Eq. (2)). Everything follows easily by induction. The most interesting case is $\varepsilon = p \cdot \varepsilon_1$. Then, by $(\mathbb{M} - \oplus)$, $(p \cdot \varepsilon_1) \oplus (p \cdot \varepsilon_1) \equiv (p + p) \cdot \varepsilon_1$ and, since the monoid is idempotent one has $(p + p) \cdot \varepsilon_1 = p \cdot \varepsilon_1$. □

Example 4.7 (*Expressions for $\mathcal{P}_\omega(\mathrm{Id})$ and 2^{Id}*)**.** The functor $\mathcal{P}_\omega(\mathrm{Id})$, which we explicitly include in our syntax of quantitative functors (since it is a non-deterministic functor), is isomorphic to the functor 2^{Id}, an instance of the monoidal exponentiation functor. We shall now show that, as expected, $\mathrm{Exp}_{\mathcal{P}_\omega(\mathrm{Id})}/_{\equiv} \cong \mathrm{Exp}_{2^{\mathrm{Id}}}/_{\equiv}$.

The expressions for $\mathcal{P}_\omega(\mathrm{Id})$ are given by the closed and guarded expressions defined in the following BNF

$$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \oplus \varepsilon \mid \mu x.\varepsilon \mid x \mid \{\varepsilon\}$$

The axioms which apply for these expressions are the axioms for fixed-points plus the axioms (*Associativity*), (*Commutativity*), (*Idempotency*) and (*Empty*).

For 2^{Id}, the expressions are given by the closed and guarded expressions defined in the following BNF

$$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \oplus \varepsilon \mid \mu x.\varepsilon \mid x \mid 1 \cdot \varepsilon \mid 0 \cdot \varepsilon$$

The axiomatization consists of the axioms for fixed-points plus (*Associativity*), (*Commutativity*), (*Empty*), $0 \cdot \varepsilon \equiv \underline{\emptyset}$ and $p \cdot \varepsilon \oplus p' \cdot \varepsilon \equiv (p + p') \cdot \varepsilon$. Because 2 is an idempotent monoid, the last axiom can be replaced, for $p = p'$, by the *(Idempotency)* axiom (by Lemma 4.6). For $p \neq p'$, note that $0 \cdot \varepsilon \equiv \underline{\emptyset}$ applies and, using the fact that $1 + 0 = 0$, the axiom $p \cdot \varepsilon \oplus p' \cdot \varepsilon \equiv (p + p) \cdot \varepsilon$ can be completely eliminated. This, together with the one but last axiom, yields that $\mathrm{Exp}_{\mathcal{P}_\omega(\mathrm{Id})}/_{\equiv} \cong \mathrm{Exp}_{2^{\mathrm{Id}}}/_{\equiv}$.

Example 4.8 (Expressions for weighted automata)**.** The syntax canonically derived from our typing system for the functor $\mathcal{W} = \mathbb{S} \times (\mathbb{S}^{\mathrm{Id}})^A$ [2] is given by the closed and guarded expressions defined by the following BNF:

$$\begin{aligned}
\varepsilon &::= \underline{\emptyset} \mid \varepsilon \oplus \varepsilon \mid x \mid \mu x.\varepsilon \mid l\langle s\rangle \mid r\langle \varepsilon'\rangle \\
\varepsilon' &::= \underline{\emptyset} \mid \varepsilon' \oplus \varepsilon' \mid a(\varepsilon'') \\
\varepsilon'' &::= \underline{\emptyset} \mid \varepsilon'' \oplus \varepsilon'' \mid s \cdot \varepsilon
\end{aligned}$$

where $s \in \mathbb{S}$ and $a \in A$. The operational semantics of these expressions is given by the function $\delta_{\mathcal{W}\lhd\mathcal{W}}$ (hereafter denoted by $\delta_{\mathcal{W}}$) which is an instance of the general definition of $\delta_{\mathcal{F}\lhd\mathcal{H}}$ above. It is given by:

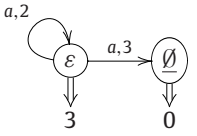
$$\begin{aligned}
\delta_{\mathcal{W}}(\underline{\emptyset}) &= \langle 0, \lambda a.\lambda\varepsilon.0\rangle \\
\delta_{\mathcal{W}}(\varepsilon_1 \oplus \varepsilon_2) &= \langle s_1 + s_2, \lambda a.\lambda\varepsilon.(f(a)(\varepsilon) + g(s)(\varepsilon))\\
&\qquad\qquad \text{where } \langle s_1, f\rangle = \delta_{\mathcal{W}}(\varepsilon_1) \text{ and } \langle s_2, g\rangle = \delta_{\mathcal{W}}(\varepsilon_2) \\
\delta_{\mathcal{W}}(\mu x.\varepsilon) &= \delta_{\mathcal{W}}(\varepsilon[\mu x.\varepsilon/x]) \\
\delta_{\mathcal{W}}(l\langle s\rangle) &= \langle s, \lambda a.\lambda\varepsilon.0\rangle \\
\delta_{\mathcal{W}}(r\langle \varepsilon'\rangle) &= \langle 0, \delta_{(\mathbb{S}^{\mathrm{Id}})^A\lhd\mathcal{W}}(\varepsilon')\rangle
\end{aligned}$$

[2] To be completely precise (in order for $\mathcal{W}$ to be a quantitative functor) here the leftmost $\mathbb{S}$ should be written as $\mathbb{S}^{\{*\}}$. However, it is easy to see that $\mathrm{Exp}_{\mathbb{S}^{\{*\}}}/_{\equiv} \cong \mathbb{S}$ and so we will omit this detail from now on.

$$
\begin{aligned}
\delta_{(\mathbb{S}^{\mathrm{Id}})^A \triangleleft \mathcal{W}}(\underline{\emptyset}) &= \lambda a.\lambda\varepsilon.0 \\
\delta_{(\mathbb{S}^{\mathrm{Id}})^A \triangleleft \mathcal{W}}(\varepsilon_1 \oplus \varepsilon_2) &= \lambda a.\lambda\varepsilon.(f(a)(\varepsilon) + g(s)(\varepsilon)) \\
&\qquad \text{where } f = \delta_{(\mathbb{S}^{\mathrm{Id}})^A \triangleleft \mathcal{W}}(\varepsilon_1) \text{ and } g = \delta_{(\mathbb{S}^{\mathrm{Id}})^A \triangleleft \mathcal{W}}(\varepsilon_2) \\
\delta_{(\mathbb{S}^{\mathrm{Id}})^A \triangleleft \mathcal{W}}(a(\varepsilon'')) &= \lambda a'. \begin{cases} \delta_{S^{\mathrm{Id}} \triangleleft \mathcal{W}}(\varepsilon'') & \text{if } a = a' \\ \lambda\varepsilon.0 & \text{otherwise} \end{cases}
\end{aligned}
$$

$$
\begin{aligned}
\delta_{(\mathbb{S}^{\mathrm{Id}}) \triangleleft \mathcal{W}}(\underline{\emptyset}) &= \lambda\varepsilon.0 \\
\delta_{(\mathbb{S}^{\mathrm{Id}}) \triangleleft \mathcal{W}}(\varepsilon_1 \oplus \varepsilon_2) &= \lambda\varepsilon.(f(\varepsilon) + g(\varepsilon)) \\
&\qquad \text{where } f = \delta_{(\mathbb{S}^{\mathrm{Id}}) \triangleleft \mathcal{W}}(\varepsilon_1) \text{ and } g = \delta_{(\mathbb{S}^{\mathrm{Id}}) \triangleleft \mathcal{W}}(\varepsilon_2) \\
\delta_{(\mathbb{S}^{\mathrm{Id}}) \triangleleft \mathcal{W}}(s \cdot \varepsilon) &= \lambda\varepsilon'. \begin{cases} s & \text{if } \varepsilon = \varepsilon' \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

The function $\delta_{\mathcal{W}}$ assigns to each expression ε a pair $\langle s, t\rangle$, consisting of an output weight $s \in \mathbb{S}$ and a function $t\colon A \to \mathbb{S}^{\mathrm{Exp}_{\mathcal{W}}}$. For a concrete example, let $\mathbb{S} = \mathbb{R}, A = \{a\}$, and consider $\varepsilon = \mu x.r\langle a(2 \cdot x \oplus 3 \cdot \underline{\emptyset})\rangle \oplus l\langle 1\rangle \oplus l\langle 2\rangle$. The semantics of this expression, obtained by $\delta_{\mathcal{W}}$ is described by the weighted automaton below.



$$o = \begin{pmatrix} 3 \\ 0 \end{pmatrix} \qquad T_a = \begin{pmatrix} 2 & 0 \\ 3 & 0 \end{pmatrix}$$

In Table 1, a more concise syntax for expressions for weighted automata is presented (together with an axiomatization). We remark that this syntax is a subset of the one proposed in [13] (there a parallel composition was also considered), but the axiomatization is new.

In order to see that the concise syntax and axiomatization are correct, one has to write translation maps between both syntaxes, which we show next, and then prove that if two expressions are provably equivalent in one syntax then their translations are provably equivalent as well. We will not show the full proof here but we will illustrate one case.

First, we translate the syntax presented in Table 1 into the canonically derived syntax presented above.

$$
\begin{aligned}
(\underline{\emptyset})^\dagger &= \underline{\emptyset} & s^\dagger &= l\langle s\rangle \\
(\varepsilon_1 \oplus \varepsilon_2)^\dagger &= (\varepsilon_1)^\dagger \oplus (\varepsilon_2)^\dagger & (a(s \cdot \varepsilon))^\dagger &= r\langle a(s \cdot \varepsilon^\dagger)\rangle \\
(\mu x.\varepsilon)^\dagger &= \mu x.\varepsilon^\dagger & x^\dagger &= x
\end{aligned}
$$

And now the converse translation:

$$
\begin{aligned}
(\underline{\emptyset})^\ddagger &= \underline{\emptyset} & (r\langle\underline{\emptyset}\rangle)^\ddagger &= \underline{\emptyset} \\
(\varepsilon_1 \oplus \varepsilon_2)^\ddagger &= (\varepsilon_1)^\ddagger \oplus (\varepsilon_2)^\ddagger & (r\langle\varepsilon_1' \oplus \varepsilon_2'\rangle)^\ddagger &= (r\langle\varepsilon_1'\rangle)^\ddagger \oplus (r\langle\varepsilon_2'\rangle)^\ddagger \\
(\mu x.\varepsilon)^\ddagger &= \mu x.\varepsilon^\ddagger & (r\langle a(\underline{\emptyset})\rangle)^\ddagger &= \underline{\emptyset} \\
x^\ddagger &= x & (r\langle a(\varepsilon_1'' \oplus \varepsilon_2'')\rangle)^\ddagger &= (r\langle a(\varepsilon_1'')\rangle)^\ddagger \oplus (r\langle a(\varepsilon_2'')\rangle)^\ddagger \\
(l\langle s\rangle)^\ddagger &= s & (r\langle a(s \cdot \varepsilon)\rangle)^\ddagger &= a(s \cdot \varepsilon^\ddagger)
\end{aligned}
$$

Let us next show an example of the correctness of the syntax and axioms presented in Table 1. Take, from Table 1, the axiom $a(0 \cdot \varepsilon) \equiv \underline{\emptyset}$. We need to prove that $(a(0 \cdot \varepsilon))^\dagger \equiv \underline{\emptyset}^\dagger$, using the canonically derived axioms for $\mathrm{Exp}_{\mathcal{W}}$. The left expression would be translated to $r\langle a(0 \cdot \varepsilon^\dagger)\rangle$, whereas $\underline{\emptyset}$ would just be translated to $\underline{\emptyset}$. Next, using the axioms of $\mathrm{Exp}_{\mathcal{W}}$ one derives $r\langle a(0 \cdot \varepsilon^\dagger)\rangle \equiv r\langle a(\underline{\emptyset})\rangle \equiv r\langle\underline{\emptyset}\rangle \equiv \underline{\emptyset}$, as expected.

We are now ready to formulate the analogue of Kleene's theorem for quantitative systems.

Theorem 4.9 (Kleene's theorem for quantitative functors)**.** *Let $\mathcal{H}$ be a quantitative functor.*

1. *For every locally finite $\mathcal{H}$-coalgebra (S, h) and for every $s \in S$ there exists an expression $\varepsilon_s \in \mathrm{Exp}_{\mathcal{H}}$ such that $s \sim_b \varepsilon_s$.*
2. *For every $\varepsilon \in \mathrm{Exp}_{\mathcal{H}}$, there exists a finite $\mathcal{H}$-coalgebra (S, h) with $s \in S$ such that $s \sim_b \varepsilon$.*

Proof. Let $\mathcal{H}$ be a quantitative functor.

Proof of item 1. Let $s \in S$ and let $\langle s \rangle = \{s_1, \ldots, s_n\}$ with $s_1 = s$. We construct, for every state $s_i \in \langle s \rangle$, an expression $\langle\!\langle\, s_i \,\rangle\!\rangle$ such that $s_i \sim \langle\!\langle\, s_i \,\rangle\!\rangle$ (and thus $s_i \sim_b \langle\!\langle\, s_i \,\rangle\!\rangle$).

If $\mathcal{H} = \mathrm{Id}$, we set, for every i, $\langle\!\langle\, s_i \,\rangle\!\rangle = \underline{\emptyset}$. It is easy to see that $\{\langle s_i, \underline{\emptyset}\rangle \mid s_i \in \langle s \rangle\}$ is a bisimulation and, thus, we have that $s \sim \langle\!\langle\, s \,\rangle\!\rangle$.

For $\mathcal{H} \neq \mathrm{Id}$, we proceed in the following way. Let, for every i, $A_i = \mu x_i.\gamma^{\mathcal{H}}_{h(s_i)}$ where, for $\mathcal{F} \lhd \mathcal{H}$ and $c \in \mathcal{F}\langle s \rangle$, the expression $\gamma^{\mathcal{F}}_c \in \mathrm{Exp}_{\mathcal{F} \lhd \mathcal{H}}$ is defined by induction on the structure of $\mathcal{F}$:

$$\gamma^{\mathrm{Id}}_{s_i} = x_i \qquad \gamma^{\mathrm{B}}_b = b \qquad \gamma^{\mathcal{F}_1 \times \mathcal{F}_2}_{\langle c,c' \rangle} = l\langle \gamma^{\mathcal{F}_1}_c \rangle \oplus r\langle \varepsilon^{\mathcal{F}_2}_{c'} \rangle \quad \gamma^{\mathcal{F}^A}_f = \bigoplus_{a \in A} a\left(\gamma^{\mathcal{F}}_{f(a)}\right)$$

$$\gamma^{\mathcal{F}_1 \oplus \mathcal{F}_2}_{\kappa_1(c)} = l[\gamma^{\mathcal{F}_1}_c] \quad \gamma^{\mathcal{F}_1 \oplus \mathcal{F}_2}_{\kappa_2(c)} = r[\gamma^{\mathcal{F}_2}_c] \quad \gamma^{\mathcal{F}_1 \oplus \mathcal{F}_2}_{\perp} = \underline{\emptyset} \qquad \gamma^{\mathcal{F}_1 \oplus \mathcal{F}_2}_{\top} = l[\underline{\emptyset}] \oplus r[\underline{\emptyset}]$$

$$\gamma^{\mathcal{P}_\omega \mathcal{F}}_C = \begin{cases} \bigoplus_{c \in C} \{\gamma^{\mathcal{F}}_c\} & C \neq \{\} \\ \underline{\emptyset} & \text{otherwise} \end{cases} \qquad \gamma^{\mathbb{M}^{\mathcal{H}_1}}_f = \bigoplus_{\substack{c \,\in\, \mathcal{H}_1(\langle s \rangle) \\ f(c) \neq 0}} f(c) \cdot \gamma^{\mathcal{H}_1}_c$$

Now, let $A^0_i = A_i$, define $A^{k+1}_i = A^k_i\{A^k_{k+1}/x_{k+1}\}$ and then set $\langle\!\langle\, s_i \,\rangle\!\rangle = A^n_i$. Here, $A\{A'/x\}$ denotes syntactic replacement (that is, substitution without renaming of bound variables in A which are also free variables in A').

Observe that the term

$$A^n_i = \left(\mu x_i.\gamma^{\mathcal{H}}_{h(s_i)}\right)\left\{A^0_1/x_1\right\}\ldots\left\{A^{n-1}_n/x_n\right\}$$

is a closed term because, for every $j = 1, \ldots, n$, the term A^{j-1}_j contains at most $n - j$ free variables in the set $\{x_{j+1}, \ldots, x_n\}$.

It remains to prove that $s_i \sim \langle\!\langle\, s_i \,\rangle\!\rangle$. We show that $R = \{\langle s_i, \langle\!\langle\, s_i \,\rangle\!\rangle\rangle \mid s_i \in \langle s \rangle\}$ is a bisimulation. For that, we first define, for $\mathcal{F} \lhd \mathcal{H}$ and $c \in \mathcal{F}\langle s \rangle$, $\xi^{\mathcal{F}}_c = \gamma^{\mathcal{F}}_c\{A^0_1/x_1\}\ldots\{A^{n-1}_n/x_n\}$ and the relation

$$R_{\mathcal{F} \lhd \mathcal{H}} = \left\{\langle c, \delta_{\mathcal{F} \lhd \mathcal{H}}(\xi^{\mathcal{F}}_c)\rangle \mid c \in \mathcal{F}\langle s \rangle\right\}.$$

Then, we prove that ① $R_{\mathcal{F} \lhd \mathcal{H}} = \overline{\mathcal{F}}(R)$ and ② $\langle h(s_i), \delta_{\mathcal{H}}(\langle\!\langle\, s_i \,\rangle\!\rangle)\rangle \in R_{\mathcal{H} \lhd \mathcal{H}}$.

① By induction on the structure of $\mathcal{F}$. We will show here only the proof for the case $\mathcal{F} = \mathbb{M}^{\mathcal{H}_1}$.

$$\begin{aligned}
&\langle f, g\rangle \in \overline{\mathbb{M}^{\mathcal{H}_1}}(R) \\
\Leftrightarrow\ &\exists_{\varphi:\, \mathcal{H}_1(R) \to \mathbb{M}}\ \mathbb{M}^{\mathcal{H}_1}(\pi_1)(\varphi) = f \text{ and } \mathbb{M}^{\mathcal{H}_1}(\pi_2)(\varphi) = g && \text{(def. } \overline{\mathbb{M}^{\mathcal{H}_1}}(R)\text{)} \\
\Leftrightarrow\ &f(u) = \sum_{\langle u,y\rangle \in \overline{\mathcal{H}_1}(R)} \varphi(\langle u, y\rangle) \text{ and } g(v) = \sum_{\langle x,v\rangle \in \overline{\mathcal{H}_1}(R)} \varphi(\langle x, v\rangle) && \text{(def. } \mathbb{M}^{\mathcal{H}_1} \text{ on arrows)} \\
\Leftrightarrow\ &f(u) = \sum_{\langle u,y\rangle \in R_{\mathcal{H}_1 \lhd \mathcal{H}}} \varphi(\langle u, y\rangle) \text{ and } g(v) = \sum_{\langle x,v\rangle \in R_{\mathcal{H}_1 \lhd \mathcal{H}}} \varphi(\langle x, v\rangle) && \text{(ind. hyp.)} \\
\Leftrightarrow\ &f(u) = \varphi\left(\langle u, \delta(\xi^{\mathcal{H}_1}_u)\rangle\right) \text{ and } g(v) = \sum_{v=\delta(\xi^{\mathcal{H}_1}_x)} \varphi\left(\langle x, \delta(\xi^{\mathcal{H}_1}_x)\rangle\right) && \text{(def. } R_{\mathcal{H}_1 \lhd \mathcal{H}}\text{)} \\
\Leftrightarrow\ &f(u) = \varphi\left(\langle u, \delta(\xi^{\mathcal{H}_1}_u)\rangle\right) \text{ and } g(v) = \sum_{v=\delta(\xi^{\mathcal{H}_1}_x)} f(x) && \text{(def. } f\text{)} \\
\Leftrightarrow\ &f \in \mathbb{M}^{\mathcal{H}_1}(\langle s \rangle) \text{ and } g = \delta_{\mathbb{M}^{\mathcal{H}_1} \lhd \mathcal{H}}\left(\bigoplus_{f(x) \neq 0} f(x) \cdot \xi^{\mathcal{H}_1}_x\right) && \text{(def. } \delta_{\mathbb{M}^{\mathcal{H}_1} \lhd \mathcal{H}}\text{)} \\
\Leftrightarrow\ &f \in \mathbb{M}^{\mathcal{H}_1}(\langle s \rangle) \text{ and } g = \delta_{\mathbb{M}^{\mathcal{H}_1} \lhd \mathcal{H}}\left(\xi^{\mathbb{M}^{\mathcal{H}_1}}_f\right) && \text{(def. } \xi^{\mathbb{M}^{\mathcal{H}_1}}_f\text{)} \\
\Leftrightarrow\ &\langle f, g\rangle \in R_{\mathbb{M}^{\mathcal{H}_1} \lhd \mathcal{H}}
\end{aligned}$$

② We want to prove that $\langle g(s_i), \delta_{\mathcal{H}}(\langle\!\langle\, s_i \,\rangle\!\rangle)\rangle \in R_{\mathcal{H} \lhd \mathcal{H}}$. For that, we must show that $g(s_i) \in \mathcal{H}\langle s \rangle$ and $\delta_{\mathcal{H}}(\langle\!\langle\, s_i \,\rangle\!\rangle) = \delta_{\mathcal{H}}(\xi^{\mathcal{H}}_{g(s_i)})$. The latter follows by definition of $\langle s \rangle$, whereas for the former we observe that:

$$\begin{aligned}
&\delta_{\mathcal{H}}(\langle\!\langle\, s_i \,\rangle\!\rangle) \\
&= \delta_{\mathcal{H}}\left(\left(\mu x_i.\gamma^{\mathcal{H}}_{g(s_i)}\right)\left\{A^0_1/x_1\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\right) && \text{(def. of } \langle\!\langle\, s_i \,\rangle\!\rangle\text{)} \\
&= \delta_{\mathcal{H}}\left(\mu x_i.\gamma^{\mathcal{H}}_{g(s_i)}\left\{A^0_1/x_1\right\}\ldots\left\{A^{i-2}_{i-1}/x_{i-1}\right\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\right)
\end{aligned}$$

$$
\begin{aligned}
&= \delta_{\mathcal{H}}\left(\gamma^{\mathcal{H}}_{g(s_i)}\left\{A^0_1/x_1\right\}\ldots\left\{A^{i-2}_{i-1}/x_{i-1}\right\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}[A^n_i/x_i]\right) \quad \text{(def. of } \delta_{\mathcal{H}}\text{)}\\
&= \delta_{\mathcal{H}}\left(\gamma^{\mathcal{H}}_{g(s_i)}\left\{A^0_1/x_1\right\}\ldots\left\{A^{i-2}_{i-1}/x_{i-1}\right\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\{A^n_i/x_i\}\right) \quad ([A^n_i/x_i]=\{A^n_i/x_i\})\\
&= \delta_{\mathcal{H}}\left(\gamma^{\mathcal{H}}_{g(s_i)}\left\{A^0_1/x_1\right\}\ldots\left\{A^{i-2}_{i-1}/x_{i-1}\right\}\{A^n_i/x_i\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\right)\\
&= \delta_{\mathcal{H}}\left(\xi^{\mathcal{H}}_{g(s_i)}\right)
\end{aligned}
$$

Here, note that $[A^n_i/x_i]=\{A^n_i/x_i\}$, because A^n_i has no free variables. The last two steps follow, respectively, because x_i is not free in $A^i_{i+1},\ldots,A^{n-1}_n$ and:

$$
\begin{aligned}
&\{A^n_i/x_i\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\\
&=\left\{A^{i-1}_i\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}/x_i\right\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\\
&=\left\{A^{i-1}_i/x_i\right\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\} \qquad (3)
\end{aligned}
$$

Eq. (3) uses the syntactic identity

$$A\{B\{C/y\}/x\}\{C/y\}=A\{B/x\}\{C/y\},\quad y \text{ not free in } C \qquad (4)$$

Proof of item 2. We want to show that for every expression $\varepsilon\in \mathrm{Exp}_{\mathcal{H}}$ there is a finite $\mathcal{H}$-coalgebra (S,f) with $s\in S$ such that $s\sim_b \varepsilon$. We construct such coalgebra in the following way.

Again, we only show the proof for $\mathcal{H}=\mathbb{M}^{\mathcal{H}_1}$. The case when $\mathcal{H}$ is a non-deterministic functor is covered by Theorem 2.12 and the other cases ($\mathbb{M}^{\mathcal{H}}\times\mathbb{M}^{\mathcal{H}}$, $(\mathbb{M}^{\mathcal{H}})^A$ and $\mathbb{M}^{\mathcal{H}}\oplus\mathbb{M}^{\mathcal{H}}$) follow directly from the $\mathcal{H}=\mathbb{M}^{\mathcal{H}_1}$, which we shall prove next.

For $\varepsilon\in\mathrm{Exp}_{\mathbb{M}^{\mathcal{H}_1}\triangleleft\mathcal{H}}$, we set $(S,h)=\langle\varepsilon\rangle$ (recall that ε is the subcoalgebra generated by ε). We now just have to prove that S is finite. In fact we shall prove that $S\subseteq cl(\varepsilon)$, where $cl(\varepsilon)$ denotes the smallest subset containing all subformulas of ε and the unfoldings of μ (sub)formulas, that is, the smallest subset satisfying:

$$
\begin{array}{llll}
cl(\underline{\emptyset}) & = \{\underline{\emptyset}\} & cl(l[\varepsilon_1]) & = \{l[\varepsilon_1]\}\cup cl(\varepsilon_1)\\
cl(\varepsilon_1\oplus\varepsilon_2) & = \{\varepsilon_1\oplus\varepsilon_2\}\cup cl(\varepsilon_1)\cup cl(\varepsilon_2) & cl(r[\varepsilon_1]) & = \{r[\varepsilon_1]\}\cup cl(\varepsilon_1)\\
cl(\mu x.\varepsilon_1) & = \{\mu x.\,\varepsilon_1\}\cup cl(\varepsilon_1[\mu x.\varepsilon_1/x]) & cl(a(\varepsilon_1)) & = \{a(\varepsilon_1)\}\cup cl(\varepsilon_1)\\
cl(l\langle\varepsilon_1\rangle) & = \{l\langle\varepsilon_1\rangle\}\cup cl(\varepsilon_1) & cl(\{\varepsilon_1\}) & = \{\{\varepsilon_1\}\}\cup cl(\varepsilon_1)\\
cl(r\langle\varepsilon_1\rangle) & = \{r\langle\varepsilon_1\rangle\}\cup cl(\varepsilon_1) & cl(m\cdot\varepsilon_1) & = \{m\cdot\varepsilon_1\}\cup cl(\varepsilon_1)
\end{array}
$$

Note that the set $cl(\varepsilon)$ is finite (the number of unfoldings is finite).

To show that $S\subseteq cl(\varepsilon)$ (S is the state space of $\langle\varepsilon\rangle$), it is enough to show that, for any $c\in\mathcal{H}_1(\mathrm{Exp}_{\mathbb{M}^{\mathcal{H}_1}\triangleleft\mathcal{H}})$, $\delta_{\mathbb{M}^{\mathcal{H}_1}\triangleleft\mathcal{H}}(\varepsilon)(c)\neq 0\Rightarrow c\in\mathcal{H}_1(cl(\varepsilon))$.

It is an easy proof by induction on the product of types of expressions and expressions (using the order defined in Eq. (2)). We exemplify the cases $\varepsilon=\varepsilon_1\oplus\varepsilon_2$

$$
\begin{aligned}
&\delta_{\mathbb{M}^{\mathcal{H}_1}\triangleleft\mathcal{H}}(\varepsilon_1\oplus\varepsilon_2)(c)\neq 0\\
\Leftrightarrow\ &\delta_{\mathbb{M}^{\mathcal{H}_1}\triangleleft\mathcal{H}}(\varepsilon_1)(c)\neq 0 \text{ or } \delta_{\mathbb{M}^{\mathcal{H}_1}\triangleleft\mathcal{H}}(\varepsilon_2)(c)\neq 0 \quad \text{(def. } \delta_{\mathbb{M}^{\mathcal{H}_1}\triangleleft\mathcal{H}}\text{)}\\
\Rightarrow\ &c\in\mathcal{H}_1(cl(\varepsilon_1)) \text{ or } c\in\mathcal{H}_1(cl(\varepsilon_2)) \qquad \text{(ind. hyp.)}\\
\Rightarrow\ &c\in\mathcal{H}_1(cl(\varepsilon_1\oplus\varepsilon_2)) \qquad \text{(def. } cl\text{)}
\end{aligned}
$$

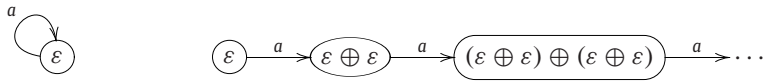
and $\varepsilon=\mu x.\varepsilon_1$

$$
\begin{aligned}
&\delta_{\mathbb{M}^{\mathcal{H}_1}}(\mu x.\varepsilon_1)(c)\neq 0\\
\Leftrightarrow\ &\delta_{\mathbb{M}^{\mathcal{H}_1}}(\varepsilon_1[\mu x.\varepsilon_1/x])(c)\neq 0 \quad \text{(def. } \delta_{\mathbb{M}^{\mathcal{H}_1}}(\mu x.\varepsilon_1)\text{)}\\
\Rightarrow\ &c\in\mathcal{H}_1(cl(\varepsilon_1[\mu x.\varepsilon_1/x])) \qquad \text{(ind. hyp.)}\\
\Rightarrow\ &c\in\mathcal{H}_1(cl(\mu x.\varepsilon_1)) \qquad (\mathcal{H}_1(cl(\varepsilon_1[\mu x.\varepsilon_1/x]))\subseteq\mathcal{H}_1(cl(\mu x.\varepsilon_1)))
\end{aligned}
$$

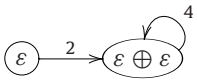
□

We can now explain the technical reason why we consider in this section only functors that are not mixed.

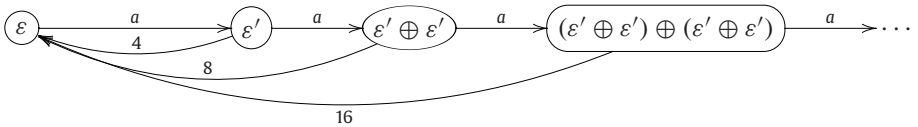
In the case of a non-deterministic functor $\mathcal{G}$, the proof of item 2. above requires considering subcoalgebras modulo (*Associativity*), (*Commutativity*) and (*Idempotency*) (ACI). Consider for instance the expression $\varepsilon = \mu x.r\langle a(x \oplus x)\rangle$ of type $\mathcal{D} = 2 \times \mathrm{Id}^A$. The subcoalgebras generated with and without applying ACI are the following:



In the case of $\mathbb{M}^{\mathcal{H}}$ (or $\mathbb{M}^{\mathcal{H}} \times \mathbb{M}^{\mathcal{H}}$, $(\mathbb{M}^{\mathcal{H}})^A$ and $\mathbb{M}^{\mathcal{H}} \oplus \mathbb{M}^{\mathcal{H}}$), the idempotency axiom does not hold anymore. However, surprisingly enough, in these cases proving the finiteness of the subcoalgebra $\langle\varepsilon\rangle$ is not problematic. The key observation is that the monoid structure will be able to avoid the infinite scenario described above. What happens is concisely captured by the following example. Take the expression $\varepsilon = \mu x.2 \cdot (x \oplus x)$ for the functor $\mathbb{R}^{\mathrm{Id}}$. Then, the subcoalgebra generated by ε is depicted in the following picture:



The syntactic restriction that excludes mixed functors is needed because of the following problem. Take as an example the functor $\mathbb{M}^{\mathrm{Id}} \times \mathrm{Id}^A$. A well-typed expression for this functor would be $\varepsilon = \mu x.r\langle a(x \oplus x \oplus l\langle 2 \cdot x\rangle \oplus l\langle 2 \cdot x\rangle)\rangle$. It is clear that we cannot apply idempotency in the subexpression $x \oplus x \oplus l\langle 2 \cdot x\rangle \oplus l\langle 2 \cdot x\rangle$ and hence the subcoalgebra generated by ε will be infinite:



with $\varepsilon' = \varepsilon \oplus \varepsilon \oplus l\langle 2 \cdot \varepsilon\rangle \oplus l\langle 2 \cdot \varepsilon\rangle$. We will show in the next section how to overcome this problem.

Let us summarize what we have achieved so far: we have presented a framework that allows, for each quantitative functor $\mathcal{H} \in QF$, the derivation of a language $\mathrm{Exp}_{\mathcal{H}}$. Moreover, Theorem 4.9 guarantees that for each expression $\varepsilon \in \mathrm{Exp}_{\mathcal{H}}$, there exists a finite $\mathcal{H}$-coalgebra (S, h) that contains a state $s \in S$ bisimilar to ε and, conversely, for each locally finite $\mathcal{H}$-coalgebra (S, h) and for every state in s there is an expression $\varepsilon_s \in \mathrm{Exp}_{\mathcal{H}}$ bisimilar to s.

Next, we show that the axiomatization is sound and complete with respect to behavioural equivalence.

4.1. Soundness and completeness

The proof of soundness and completeness follows exactly the same structure as the ones presented in [8] for non-deterministic functors (with the difference that now we use behavioural equivalence $\sim_b$ instead of bisimilarity $\sim$). We will recall all the steps here, but will only show the proof of each theorem and lemma for the new case of the monoidal exponentiation functor.

The relation $\equiv$ gives rise to the equivalence map $[-]\colon \mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}} \to \mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}/_{\equiv}$, defined by $[\varepsilon] = \{\varepsilon' \mid \varepsilon \equiv \varepsilon'\}$. The following diagram summarizes the maps we have defined so far:

$$\begin{array}{ccc} \mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}} & \xrightarrow{[-]} & \mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}/_{\equiv} \\ {\scriptstyle \delta_{\mathcal{F}\triangleleft\mathcal{H}}}\downarrow & & \\ F(\mathrm{Exp}_{\mathcal{H}}) & \xrightarrow[\mathcal{F}[-]]{} & \mathcal{F}(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}) \end{array}$$

In order to complete the diagram, we next prove that the relation $\equiv$ is contained in the kernel of $\mathcal{F}[-] \circ \delta_{\mathcal{F}\triangleleft\mathcal{H}}$[3]. This will guarantee the existence of a well-defined function $\partial_{\mathcal{F}\triangleleft\mathcal{H}}\colon \mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}/_{\equiv} \to \mathcal{F}(\mathrm{Exp}_{\mathcal{H}}/_{\equiv})$ which, when $\mathcal{F} = \mathcal{H}$, provides $\mathrm{Exp}_{\mathcal{H}}/_{\equiv}$ with a coalgebraic structure $\partial_{\mathcal{H}}\colon \mathrm{Exp}_{\mathcal{H}}/_{\equiv} \to H(\mathrm{Exp}_{\mathcal{H}}/_{\equiv})$ (as before we write $\partial_{\mathcal{H}}$ for $\partial_{\mathcal{H}\triangleleft\mathcal{H}}$) and which makes $[-]$ a homomorphism of coalgebras.

Lemma 4.10. *Let $\mathcal{H}$ and $\mathcal{F}$ be quantitative functors, with $\mathcal{F} \triangleleft \mathcal{H}$. For all $\varepsilon_1, \varepsilon_2 \in \mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}$ with $\varepsilon_1 \equiv \varepsilon_2$,*

$$(\mathcal{F}[-]) \circ \delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon_1) = (\mathcal{F}[-]) \circ \delta_{\mathcal{F}\triangleleft\mathcal{H}}(\varepsilon_2)$$

Proof. By induction on the length of derivations of $\equiv$.

[3] This is equivalent to proving that $\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}/_{\equiv}$, together with $[-]$, is the coequalizer of the projection morphisms from $\equiv$ to $\mathrm{Exp}_{\mathcal{F}\triangleleft\mathcal{H}}$.

We just show the proof for the axioms $0 \cdot \varepsilon = \underline{\emptyset}$ and $(m \cdot \varepsilon) \oplus (m' \cdot \varepsilon) = (m + m') \cdot \varepsilon$.

$$\begin{aligned}
\delta_{\mathbb{M}^{\mathcal{H}_1 \lhd \mathcal{H}}}(0 \cdot \varepsilon) = \lambda c.0 \qquad &= \delta_{\mathbb{M}^{\mathcal{H}_1 \lhd \mathcal{H}}}(\underline{\emptyset}) \\
\delta_{\mathbb{M}^{\mathcal{H}_1 \lhd \mathcal{H}}}((m \cdot \varepsilon) \oplus (m' \cdot \varepsilon)) &= \lambda c.\delta_{\mathbb{M}^{\mathcal{H}_1 \lhd H}}(m \cdot \varepsilon)(c) + \delta_{\mathbb{M}^{\mathcal{H}_1 \lhd H}}(m' \cdot \varepsilon)(c) \\
&= \lambda c. \begin{cases} m + m' & \text{if } \delta_{\mathcal{H}_1 \lhd \mathcal{H}}(\varepsilon) = c \\ 0 & \text{otherwise} \end{cases} \\
&= \delta_{\mathbb{M}^{\mathcal{H}_1 \lhd \mathcal{H}}}((m + m') \cdot \varepsilon) \qquad \square
\end{aligned}$$

Thus, we have now provided the set $\mathrm{Exp}_{\mathcal{H}}/_{\equiv}$ with a coalgebraic structure: we have defined a function $\partial_{\mathcal{H}} : \mathrm{Exp}_{\mathcal{H}}/_{\equiv} \to H(\mathrm{Exp}_{\mathcal{H}}/_{\equiv})$, with $\partial_{\mathcal{H}}([\varepsilon]) = (H[-]) \circ \delta_{\mathcal{H}}(\varepsilon)$.

At this point we can prove soundness, since it is a direct consequence of the fact that the equivalence map $[-]$ is a coalgebra homomorphism.

Theorem 4.11 (Soundness)**.** *Let $\mathcal{H}$ be a quantitative functor. For all $\varepsilon_1, \varepsilon_2 \in \mathrm{Exp}_{\mathcal{H}}$,*

$$\varepsilon_1 \equiv \varepsilon_2 \Rightarrow \varepsilon_1 \sim_b \varepsilon_2$$

Proof. Let $\mathcal{H}$ be a quantitative functor, let $\varepsilon_1, \varepsilon_2 \in \mathrm{Exp}_{\mathcal{H}}$ and suppose that $\varepsilon_1 \equiv \varepsilon_2$. Then, $[\varepsilon_1] = [\varepsilon_2]$ and, thus

$$\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}([\varepsilon_1]) = \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}([\varepsilon_2])$$

where $\mathbf{beh}_S$ denotes, for any $\mathcal{H}$-coalgebra (S, f), the unique map into the final coalgebra. The uniqueness of the map into the final coalgebra and the fact that $[-]$ is a coalgebra homomorphism implies that

$$\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}} \circ [-] = \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}} \tag{5}$$

which then yields

$$\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}}(\varepsilon_1) = \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}}(\varepsilon_2)$$

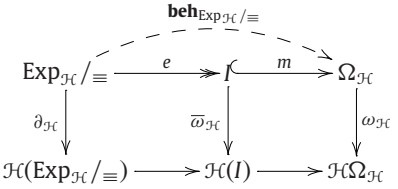
Hence, $\varepsilon_1 \sim_b \varepsilon_2$. $\square$

For completeness a bit more of work is required. Let us explain upfront the key ingredients of the proof. The goal is to prove that $\varepsilon_1 \sim_b \varepsilon_2 \Rightarrow \varepsilon_1 \equiv \varepsilon_2$. First, note that, using Eq. (5) above, we have

$$\varepsilon_1 \sim_b \varepsilon_2 \Leftrightarrow \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}}(\varepsilon_1) = \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}}(\varepsilon_2) \Leftrightarrow \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}([\varepsilon_1]) = \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}([\varepsilon_2]) \tag{6}$$

We then prove that $\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}$ is injective, which is a sufficient condition to guarantee that $\varepsilon_1 \equiv \varepsilon_2$ (since it implies, together with (6), that $[\varepsilon_1] = [\varepsilon_2]$).

We proceed as follows. First, we factorize $\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}$ into an epimorphism and a monomorphism [35, Theorem 7.1] as shown in the following diagram (where $I = \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}(\mathrm{Exp}_{\mathcal{H}}/_{\equiv})$):



(7)

Then, we prove that (1) $(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$ is a locally finite coalgebra (Lemma 4.12) and (2) both coalgebras $(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$ and $(I, \overline{\omega}_{\mathcal{H}})$ are final in the category of locally finite $\mathcal{H}$-coalgebras (Lemmas 4.15 and 4.16, respectively). Since final coalgebras are unique up to isomorphism, it follows that $e : \mathrm{Exp}_{\mathcal{H}}/_{\equiv} \to I$ is in fact an isomorphism and therefore $\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}$ is injective, which will give us completeness.

We now proceed with presenting and proving the extra lemmas needed in order to prove completeness. We start by showing that the coalgebra $(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$ is locally finite (note that this implies that $(I, \overline{\omega}_{\mathcal{H}})$ is also locally finite) and that $\partial_{\mathcal{H}}$ is an isomorphism.

Lemma 4.12. *The coalgebra $(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$ is a locally finite coalgebra. Moreover, $\partial_{\mathcal{H}}$ is an isomorphism.*

Proof. Locally finiteness is a direct consequence of the generalized Kleene's theorem (Theorem 4.9). In the proof of Theorem 4.9 we showed that given $\varepsilon \in \mathrm{Exp}_{\mathcal{H}}$, for $\mathcal{H} = \mathbb{M}^{\mathcal{H}_1}$, $\mathcal{H} = \mathbb{M}^{\mathcal{H}_1} \times \mathbb{M}^{\mathcal{H}_2}$, $\mathcal{H} = (\mathbb{M}^{\mathcal{H}_1})^A$ or $\mathcal{H} = \mathbb{M}^{\mathcal{H}_1} \oplus \mathbb{M}^{\mathcal{H}_2}$, the subcoalgebra $\langle \varepsilon \rangle$ is finite. In case $\mathcal{H}$ is a non-deterministic functor, we proved in [8] that the subcoalgebra $\langle [\varepsilon]_{ACIE} \rangle$ is finite.

Thus, the subcoalgebra $\langle[\varepsilon]\rangle$ is always finite (since $\mathrm{Exp}_{\mathcal{H}}/_{\equiv}$ is a quotient of both $\mathrm{Exp}_{\mathcal{H}}$ and $\mathrm{Exp}_{\mathcal{H}}/_{\equiv_{ACIE}}$). Recall that *ACIE* abbreviates the axioms (*Associativity*), (*Commutativity*), (*Idempotency*) and (*Empty*).

To see that $\partial_{\mathcal{H}}$ is an isomorphism, first define, for every $\mathcal{F} \lhd \mathcal{H}$,

$$\partial^{-1}_{\mathcal{F}\lhd\mathcal{H}}(c) = [\overline{\gamma}^{\mathcal{F}}_{c}] \tag{8}$$

where $\overline{\gamma}^{\mathcal{F}}_{c}$ is defined, for $\mathcal{F} \neq \mathrm{Id}$, as $\gamma^{\mathcal{F}}_{c}$ in the proof of Theorem 4.9, and for $\mathcal{F} = \mathrm{Id}$ as $\overline{\gamma}^{\mathrm{Id}}_{[\varepsilon]} = \varepsilon$. Then, we prove that the function $\partial^{-1}_{\mathcal{F}\lhd\mathcal{H}}$ has indeed the desired properties ① $\partial^{-1}_{\mathcal{F}\lhd\mathcal{H}} \circ \partial_{\mathcal{F}\lhd\mathcal{H}} = id_{\mathrm{Exp}_{\mathcal{F}\lhd\mathcal{H}}/_{\equiv}}$ and ② $\partial_{\mathcal{F}\lhd\mathcal{H}} \circ \partial^{-1}_{\mathcal{F}\lhd\mathcal{H}} = id_{\mathcal{F}(\mathrm{Exp}_{\mathcal{F}\lhd\mathcal{H}}/_{\equiv})}$. Instantiating $\mathcal{F} = \mathcal{H}$ one derives that $\delta_{\mathcal{H}}$ is an isomorphism. It is enough to prove for ① that $\overline{\gamma}^{\mathcal{F}}_{\partial_{\mathcal{F}\lhd\mathcal{H}}([\varepsilon])} \equiv \varepsilon$ and for ② that $\partial_{\mathcal{F}\lhd\mathcal{H}}([\overline{\gamma}^{\mathcal{F}}_{c}]) = c$. We just illustrate the case $\mathcal{F} = \mathbb{M}^{\mathcal{H}_1}$.

① By induction on the product of types of expressions and expressions (using the order defined in Eq. (2)).

$$\begin{aligned}
&\overline{\gamma}^{\mathbb{M}^{\mathcal{H}_1}}_{\partial_{\mathbb{M}^{\mathcal{H}_1}\lhd\mathcal{H}}([m\cdot\varepsilon])}\\
&= \bigoplus\{\partial_{\mathbb{M}^{\mathcal{H}_1}\lhd\mathcal{H}}([m\cdot\varepsilon])(c)\cdot\overline{\gamma}^{\mathcal{H}_1}_{c} \mid c \in \mathcal{H}_1(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}),\ \partial_{\mathbb{M}^{\mathcal{H}_1}\lhd\mathcal{H}}([m\cdot\varepsilon])(c) \neq 0\}\\
&= m\cdot\overline{\gamma}^{\mathcal{H}_1}_{\partial_{\mathcal{H}_1\lhd\mathcal{H}}([\varepsilon])}\\
&\equiv m\cdot\varepsilon
\end{aligned}$$

In the last step, we used the induction hypothesis, whereas in the one but last step we used the fact that $\partial_{\mathbb{M}^{\mathcal{H}_1}\lhd\mathcal{H}}([m\cdot\varepsilon])(c) \neq 0 \Leftrightarrow c = \partial_{\mathcal{H}_1\lhd\mathcal{H}}([\varepsilon])$.

② By induction on the structure of $\mathcal{F}$.

$$\begin{aligned}
\partial_{\mathbb{M}^{\mathcal{H}_1}\lhd\mathcal{H}}\left([\overline{\gamma}^{\mathbb{M}^{\mathcal{H}_1}}_{f}]\right) &= \partial_{\mathbb{M}^{\mathcal{H}_1}\lhd\mathcal{H}}([\textstyle\bigoplus\{f(c)\cdot\overline{\gamma}^{\mathcal{H}_1}_{c} \mid c \in \mathcal{H}_1(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}), f(c)\neq 0\}])\\
&= \lambda c'.\textstyle\sum\{f(c) \mid c \in \mathcal{H}_1(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}) \text{ and } c' = \partial_{\mathcal{H}_1\lhd\mathcal{H}}(\overline{\gamma}^{\mathcal{H}_1}_{c})\}\\
&\overset{IH}{=} \lambda c'.\textstyle\sum\{f(c) \mid c \in \mathcal{H}_1(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}) \text{ and } c' = c\}\\
&= f \qquad \square
\end{aligned}$$

We now prove the analogue of the following useful and intuitive equality on regular expressions, which we will then make use of to prove that there exists a coalgebra homomorphism between any locally finite coalgebra (S, h) and $(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$.

Given a deterministic automaton $\langle o, t\rangle : S \to 2 \times S^A$ and a state $s \in S$, the associated regular expression r_s can be written as

$$r_s = o(s) + \sum_{a\in A} a\cdot r_{t(s)(a)} \tag{9}$$

using the axioms of Kleene algebra [11, Theorem 4.4].

Lemma 4.13. *Let (S, h) be a locally finite $\mathcal{H}$-coalgebra, with $\mathcal{H} \neq \mathrm{Id}$, and let $s \in S$, with $\langle s\rangle = \{s_1, \ldots, s_n\}$ (where $s_1 = s$). Then:*

$$\langle\!\langle\, s_i \,\rangle\!\rangle \equiv \gamma^{\mathcal{H}}_{g(s_i)}\{\langle\!\langle\, s_1 \,\rangle\!\rangle/x_1\}\ldots\{\langle\!\langle\, s_n \,\rangle\!\rangle/x_n\} \tag{10}$$

Proof. Let A^k_i, where i and k range from 1 to n, be the terms defined as in the proof of Theorem 4.9. Recall that $\langle\!\langle\, s_i \,\rangle\!\rangle = A^n_i$. We calculate:

$$\begin{aligned}
&\langle\!\langle\, s_i \,\rangle\!\rangle\\
&= A^n_i\\
&= \left(\mu x_i.\gamma^{\mathcal{G}}_{g(s_i)}\right)\left\{A^0_1/x_1\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\\
&= \mu x_i.\left(\gamma^{\mathcal{G}}_{g(s_i)}\left\{A^0_1/x_1\right\}\ldots\left\{A^{i-2}_{i-1}/x_{i-2}\right\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\right)\\
&\equiv \gamma^{\mathcal{G}}_{g(s_i)}\{A^0_1/x_1\}\ldots\left\{A^{i-2}_{i-1}/x_{i-2}\right\}\left\{A^i_{i+1}/x_{i+1}\right\}\ldots\left\{A^{n-1}_n/x_n\right\}\{A^n_i/x_i\} \quad \text{(fixed-point axiom}^{4}\text{)}\\
&= \gamma^{\mathcal{G}}_{g(s_i)}\left\{A^0_1/x_1\right\}\ldots\left\{A^{n-1}_n/x_n\right\} \qquad \text{(by 3)}
\end{aligned}$$

[3] Note that the fixed point axiom can be formulated using syntactic replacement rather than substitution – $\gamma\{\mu x.\gamma/x\} \equiv \mu x.\gamma$ – since $\mu x.\gamma$ is a closed term.

$$
\begin{aligned}
&= \gamma^{\mathcal{G}}_{g(s_i)}\left\{A^0_1\left\{A^1_2/x_2\right\}\ldots\left\{A^{n-1}_n/x_n\right\}/x_1\right\}\ldots\left\{A^{n-1}_n/x_n\right\} && \text{(by 4)}\\
&= \gamma^{\mathcal{G}}_{g(s_i)}\left\{A^n_1/x_1\right\}\left\{A^1_2/x_2\right\}\ldots\left\{A^{n-1}_n/x_n\right\} && \text{(def. } A^n_1\text{)}\\
&\vdots && \left(\text{last 2 steps for } A^1_2,\ldots,A^{n-2}_{n-1}\right)\\
&= \gamma^{\mathcal{G}}_{g(s_i)}\left\{A^n_1/x_1\right\}\left\{A^n_2/x_2\right\}\ldots\left\{A^n_n/x_n\right\} && \left(A^n_{n-1}=A^n_n\right)\\
&= \gamma^{\mathcal{G}}_{g(s_i)}\left[A^n_1/x_1\right]\left[A^n_2/x_2\right]\ldots\left[A^n_n/x_n\right] && \text{(all } A^n_i \text{ are closed)} \qquad \square
\end{aligned}
$$

Instantiating (10) for $\langle o, t\rangle : S \to 2 \times S^A$, one can easily spot the similarity with Eq. (9) above:

$$\langle\!\langle\, s \,\rangle\!\rangle \equiv l\langle o(s)\rangle \oplus r\Big\langle \bigoplus_{a\in A} a(\langle\!\langle\, t(s)(a) \,\rangle\!\rangle)\Big\rangle$$

The above equality is used to prove that there exists a coalgebra homomorphism between any locally finite coalgebra (S, h) and $(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$.

Lemma 4.14. *Let (S, h) be a locally finite $\mathcal{H}$-coalgebra. There exists a coalgebra homomorphism $\lceil - \rceil : S \to \mathrm{Exp}_{\mathcal{H}}/_{\equiv}$.*

Proof. We define $\lceil - \rceil = [-] \circ \langle\!\langle - \rangle\!\rangle$, where $\langle\!\langle - \rangle\!\rangle$ is as in the proof of Theorem 4.9, associating to a state s of a locally finite coalgebra an expression $\langle\!\langle\, s \,\rangle\!\rangle$ with $s \sim \langle\!\langle\, s \,\rangle\!\rangle$. To prove that $\lceil - \rceil$ is a homomorphism we need to verify that $(\mathcal{H}\lceil - \rceil) \circ h = \partial_{\mathcal{H}} \circ \lceil - \rceil$.

If $\mathcal{H} = \mathrm{Id}$, then $(\mathcal{H}\lceil - \rceil) \circ g(s_i) = [\underline{\emptyset}] = \partial_{\mathcal{H}}(\lceil s_i \rceil)$. For $\mathcal{H} \neq \mathrm{Id}$ we calculate, using Lemma 4.13:

$$\partial_{\mathcal{H}} \circ \lceil s_i \rceil = \partial_{\mathcal{H}}\left(\left[\gamma^{\mathcal{H}}_{g(s_i)}[\langle\!\langle\, s_1 \,\rangle\!\rangle/x_1]\ldots[\langle\!\langle\, s_n \,\rangle\!\rangle/x_n]\right]\right)$$

and we then prove the more general equality, for $\mathcal{F} \lhd \mathcal{H}$ and $c \in \mathcal{F}\langle s\rangle$:

$$\partial_{\mathcal{F}\lhd\mathcal{H}}\left(\left[\gamma^{\mathcal{F}}_c[\langle\!\langle\, s_1 \,\rangle\!\rangle/x_1]\ldots[\langle\!\langle\, s_n \,\rangle\!\rangle/x_n]\right]\right) = \mathcal{F}\lceil - \rceil(c) \tag{11}$$

The intended equality then follows by taking $\mathcal{F} = \mathcal{H}$ and $c = g(s_i)$. Let us prove the Eq. (11) by induction on $\mathcal{F}$. We only show the case $\mathcal{F} = \mathbb{M}^{\mathcal{H}_1}$.

$$
\begin{aligned}
&\partial_{\mathbb{M}^{\mathcal{H}_1}\lhd\mathcal{H}}\left(\left[\gamma^{\mathbb{M}^{\mathcal{H}_1}}_f[\langle\!\langle\, s_1 \,\rangle\!\rangle/x_1]\ldots[\langle\!\langle\, s_n \,\rangle\!\rangle/x_n]\right]\right)\\
&= \partial_{\mathbb{M}^{\mathcal{H}_1}\lhd\mathcal{H}}([\textstyle\bigoplus\{f(c)\cdot\gamma^{\mathcal{H}_1}_c[\langle\!\langle\, s_1 \,\rangle\!\rangle/x_1]\ldots[\langle\!\langle\, s_n \,\rangle\!\rangle/x_n] \mid c \in \mathcal{H}_1(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}), f(c)\neq 0\}])\\
&= \lambda c'.\textstyle\sum\{f(c) \mid c \in \mathcal{H}_1(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}) \text{ and } c' = \partial_{\mathcal{H}_1\lhd\mathcal{H}}(\gamma^{\mathcal{H}_1}_c[\langle\!\langle\, s_1 \,\rangle\!\rangle/x_1]\ldots[\langle\!\langle\, s_n \,\rangle\!\rangle/x_n])\}\\
&\overset{IH}{=} \lambda c'.\textstyle\sum\{f(c) \mid c \in \mathcal{H}_1(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}) \text{ and } c' = (\mathcal{H}_1\lceil - \rceil)(c)\}\\
&= \mathbb{M}^{\mathcal{H}_1}(\lceil - \rceil)(f) \qquad \square
\end{aligned}
$$

We can now prove that the coalgebras $(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$ and $(I, \overline{\omega}_{\mathcal{H}})$ are both final in the category of locally finite $\mathcal{H}$-coalgebras.

Lemma 4.15. *The coalgebra $(I, \overline{\omega}_{\mathcal{H}})$ is final in the category $\mathrm{Coalg}(\mathcal{H})_{\mathbf{LF}}$.*

Proof. We want to show that for any locally finite coalgebra (S, h), there exists a *unique* homomorphism $(S, h) \to (I, \overline{\omega}_{\mathcal{H}})$. The existence is guaranteed by Lemma 4.14, where $\lceil - \rceil : S \to \mathrm{Exp}_{\mathcal{H}}/_{\equiv}$ is defined. Postcomposing this homomorphism with e (defined above, in diagram 7) we get a coalgebra homomorphism $e \circ \lceil - \rceil : S \to I$. If there is another homomorphism $f : S \to I$, then by postcomposition with the inclusion $m : I \hookrightarrow \Omega$ we get two homomorphisms ($m \circ f$ and $m \circ e \circ \lceil - \rceil$) into the final $\mathcal{H}$-coalgebra. Thus, f and $e \circ \lceil - \rceil$ must be equal. $\square$

Lemma 4.16. *The coalgebra $(\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$ is final in the category $\mathrm{Coalg}(\mathcal{H})_{\mathbf{LF}}$.*

Proof. We want to show that for any locally finite coalgebra (S, h), there exists a *unique* homomorphism $(S, h) \to (\mathrm{Exp}_{\mathcal{H}}/_{\equiv}, \partial_{\mathcal{H}})$. We only need to prove uniqueness, since the existence is guaranteed by Lemma 4.14, where $\lceil - \rceil : S \to \mathrm{Exp}_{\mathcal{H}}/_{\equiv}$ is defined.

Suppose we have another homomorphism $f : S \to \mathrm{Exp}_{\mathcal{H}}/_{\equiv}$. Then, we shall prove that $f = \lceil - \rceil$. First, observe that because f is a homomorphism the following holds for every $s \in S$ (without any risk of confusion, in this proof that follows

we denote equivalence classes $[\varepsilon]$ by expressions ε representing them):

$$f(s) = \partial_{\mathcal{H}}^{-1} \circ \mathcal{H}f \circ h(s) = \left[\gamma^{\mathcal{H}}_{g(s)}[f(s_1)/x_1] \ldots [f(s_n)/x_n]\right] \tag{12}$$

where $\langle s \rangle = \{s_1, \ldots, s_n\}$, with $s_1 = s$ (recall that $\partial_{\mathcal{H}}^{-1}$ was defined in (8) and note that $\overline{\gamma}^{\mathcal{H}}_{\mathcal{H}f \circ h(s)} = \gamma^{\mathcal{H}}_{h(s)}[f(s_i)/x_i]$).

We now have to prove that $f(s_i) = \lceil s_i \rceil$, for all $i = 1, \ldots, n$. The proof, which relies mainly on uniqueness of fixed points, is exactly the same as in the analogue theorem for non-deterministic functors and we omit it here. □

Because final objects are unique up-to isomorphism, we know that $e : \mathrm{Exp}_{\mathcal{H}}/_{\equiv} \to I$ is an isomorphism and hence we can conclude that the map $\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}$ is injective, since it factorizes into an isomorphism followed by a mono. This fact is the last ingredient we need to prove completeness.

Theorem 4.17 (Completeness). *Let $\mathcal{H}$ be a quantitative functor. For all $\varepsilon_1, \varepsilon_2 \in \mathrm{Exp}_{\mathcal{H}}$,*

$$\varepsilon_1 \sim_b \varepsilon_2 \Rightarrow \varepsilon_1 \equiv \varepsilon_2$$

Proof. Let $\mathcal{H}$ be a quantitative functor, let $\varepsilon_1, \varepsilon_2 \in \mathrm{Exp}_{\mathcal{H}}$ and suppose that $\varepsilon_1 \sim_b \varepsilon_2$, that is, $\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}}(\varepsilon_1) = \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}}(\varepsilon_2)$. Since the equivalence class map $[-]$ is a homomorphism, it holds that $\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}([\varepsilon_1]) = \mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}([\varepsilon_2])$. Now, because $\mathbf{beh}_{\mathrm{Exp}_{\mathcal{H}}/_{\equiv}}$ is injective we have that $[\varepsilon_1] = [\varepsilon_2]$. Hence, $\varepsilon_1 \equiv \varepsilon_2$. □

5. Extending the class of functors

In the previous section, we introduced regular expressions for the class of quantitative functors. In this section, by employing standard results from the theory of coalgebras, we show how to use such regular expressions to describe the coalgebras of many more functors, including the mixed functors we mentioned in Section 4.

Given $\mathcal{F}$ and $\mathcal{G}$ two endofunctors on **Set**, a *natural transformation* $\alpha : \mathcal{F} \Rightarrow \mathcal{G}$ is a family of functions $\alpha_S : \mathcal{F}(S) \to \mathcal{G}(S)$ (for all sets S), such that for all functions $h : T \to U$, $\alpha_U \circ \mathcal{F}(h) = \mathcal{G}(h) \circ \alpha_T$. If all the α_S are injective, then we say that α is injective.

Proposition 5.1. *An injective natural transformation $\alpha : \mathcal{F} \Rightarrow \mathcal{G}$ induces a functor $\alpha \circ (-) : \mathit{Coalg}(\mathcal{F})_{\mathbf{LF}} \to \mathit{Coalg}(\mathcal{G})_{\mathbf{LF}}$ that preserves and reflects behavioural equivalence.*

Proof. It is well known from [5,35] that, an injective $\alpha : \mathcal{F} \Rightarrow \mathcal{G}$ induces a functor $\alpha \circ (-) : \mathit{Coalg}(\mathcal{F}) \to \mathit{Coalg}(\mathcal{G})$ that preserves and reflects behavioural equivalence. Thus we have only to prove that $\alpha \circ (-)$ maps locally finite $\mathcal{F}$-coalgebras into locally finite $\mathcal{G}$-coalgebras.

Recall that $\alpha \circ (-)$ maps each $\mathcal{F}$-coalgebra (S, f) into the $\mathcal{G}$-coalgebra $(S, \alpha \circ f)$, and each $\mathcal{F}$-homomorphism into itself. We prove that if (S, f) is locally finite, then also $(S, \alpha \circ f)$ is locally finite.

An $\mathcal{F}$-coalgebra (S, f) is locally finite if for all $s \in S$, there exists a finite set $V_s \subseteq S$, such that $s \in V_s$ and V_s is a subsystem of S. That is, there exists a function $v : V_s \to \mathcal{F}(V_s)$, such that the inclusion $i : V_s \to S$ is an $\mathcal{F}$-homomorphism between (V_s, v) and (S, f). At this point, note that if $i : V \to S$ is an $\mathcal{F}$-homomorphism between (V_s, v) and (S, f), then it is also a $\mathcal{G}$-homomorphism between $(V_s, \alpha \circ v)$ and $(S, \alpha \circ f)$. Thus, $(S, \alpha \circ f)$ is locally finite. □

This result allows us to extend our framework to many other functors, as we shall explain next. Consider a functor $\mathcal{F}$ which is not quantitative and suppose there exists an injective natural transformation α from $\mathcal{F}$ into some quantitative functor $\mathcal{H}$. A (locally finite) $\mathcal{F}$-coalgebra can be translated into a (locally finite) $\mathcal{H}$-coalgebra via the functor $\alpha \circ (-)$ and then it can be characterized by using expressions in $\mathrm{Exp}_{\mathcal{H}}$. The axiomatization for $\mathrm{Exp}_{\mathcal{H}}$ is still sound and complete for $\mathcal{F}$-coalgebras, since the functor $\alpha \circ (-)$ preserves and reflects behavioural equivalence.

However, note that (half of) Kleene's theorem does not hold anymore, because not all the expressions in $\mathrm{Exp}_{\mathcal{H}}$ denote $\mathcal{F}$-behaviours or, more precisely, not all expressions in $\mathrm{Exp}_{\mathcal{H}}$ are equivalent to $\mathcal{H}$-coalgebras that are in the image of $\alpha \circ (-)$. Thus, in order to retrieve Kleene's theorem, one has to exclude such expressions. In many situations, this is feasible by simply imposing some syntactic constraints on $\mathrm{Exp}_{\mathcal{H}}$.

Let us illustrate this by means of an example. First, we recall the definition of the probability functor (which will play a key role in the next section).

Definition 5.2 (*Probability functor*). A probability distribution over a set S is a function $d : S \to [0, 1]$ such that $\sum_{s \in S} d(s) = 1$. The probability functor $\mathcal{D}_\omega :$ **Set** $\to$ **Set** is defined as follows. For all sets S, $\mathcal{D}_\omega(S)$ is the set of probability distributions over S with finite support. For all functions $h : S \to T$, $\mathcal{D}_\omega(h)$ maps each $d \in \mathcal{D}_\omega(S)$ into d^h (Definition 3.1).

Note that for any set S, $\mathcal{D}_\omega(S) \subseteq \mathbb{R}^S$ since probability distributions are also functions from S to $\mathbb{R}$. Let ι be the family of inclusions $\iota_S : \mathcal{D}_\omega(S) \to \mathbb{R}^S$. It is easy to see that ι is a natural transformation between $\mathcal{D}_\omega$ and $\mathbb{R}^{\mathrm{Id}}$ (the two functors are defined in the same way on arrows). Thus, in order to specify $\mathcal{D}_\omega$-coalgebras, we will use $\varepsilon \in \mathrm{Exp}_{\mathbb{R}^{\mathrm{Id}}}$. These are the closed

and guarded expressions given by the following BNF, where $r \in \mathbb{R}$ and $x \in X$ (X a set of fixed-point variables)

$$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \oplus \varepsilon \mid x \mid \mu x.\varepsilon \mid r \cdot \varepsilon$$

This language is enough to specify all $\mathcal{D}_\omega$-behaviours, but it also allows us to specify $\mathbb{R}^{\mathrm{Id}}$-behaviours that are not $\mathcal{D}_\omega$-behaviours, such as for example, $\mu x.2 \cdot x$ and $\mu x.0 \cdot x$. In order to obtain a language $\mathrm{Exp}_{\mathcal{D}_\omega}$ that specifies all and only the regular $\mathcal{D}_\omega$-behaviours, it suffices to change the BNF above as follows:

$$\varepsilon ::= x \mid \mu x.\varepsilon \mid \bigoplus_{i\in 1,\dots,n} p_i \cdot \varepsilon_i \quad \text{for } p_i \in (0,1] \text{ such that } \sum_{i\in 1,\dots,n} p_i = 1 \tag{13}$$

where $\bigoplus_{i\in 1,\dots,n} p_i \cdot \varepsilon_i$ denotes $p_1 \cdot \varepsilon_1 \oplus \cdots \oplus p_n \cdot \varepsilon_n$.

Next, we prove Kleene's theorem for this restricted syntax. Note that the procedure of appropriately restricting the syntax usually requires some ingenuity. We shall see that in many concrete cases, as for instance $\mathcal{D}_\omega$ above, it is fairly intuitive which restriction to choose. Also, although we cannot provide a uniform proof of Kleene's theorem, the proof for each concrete example is a slight adaptation of the more general one (Theorem 4.9).

Theorem 5.3 (Kleene's Theorem for the probability functor)**.**

1. *For every locally finite $\mathcal{D}_\omega$-coalgebra (S, d) and for every $s \in S$ there exists an expression $\varepsilon_s \in \mathrm{Exp}_{\mathcal{D}_\omega}$ such that $s \sim_b \varepsilon_s$.*
2. *For every $\varepsilon \in \mathrm{Exp}_{\mathcal{D}_\omega}$, there exists a finite $\mathcal{D}_\omega$-coalgebra (S, d) with $s \in S$ such that $s \sim_b \varepsilon$.*

Proof. Let $s \in S$ and let $\langle s\rangle = \{s_1, \dots, s_n\}$ with $s_1 = s$. We construct, for every state $s_i \in \langle s\rangle$, an expression $\langle\!\langle\, s_i \,\rangle\!\rangle$ such that $s_i \sim \langle\!\langle\, s_i \,\rangle\!\rangle$ (and thus $s \sim_b \varepsilon_s$) .

Let, for every i, $A_i = \mu x_i. \bigoplus_{d(s_i)(s_j)\neq 0} d(s_i)(s_j) \cdot x_j$.

Now, let $A_i^0 = A_i$, define $A_i^{k+1} = A_i^k\{A_{k+1}^k/x_{k+1}\}$ and then set $\langle\!\langle\, s_i \,\rangle\!\rangle = A_i^n$. Observe that the term

$$A_i^n = \left(\mu x_i. \bigoplus_{d(s_i)(s_j)\neq 0} d(s_i)(s_j) \cdot x_j\right)\left\{A_1^0/x_1\right\}\dots\left\{A_n^{n-1}/x_n\right\}$$

is a closed term and $\sum_{d(s_i)(s_j)\neq 0} d(s_i)(s_j) = 1$. Thus, $A_i^n \in \mathrm{Exp}_{\mathcal{D}_\omega}$.

It remains to prove that $s_i \sim_b \langle\!\langle\, s_i \,\rangle\!\rangle$. We show that $R = \{\langle s_i, \langle\!\langle\, s_i \,\rangle\!\rangle\rangle \mid s_i \in \langle s\rangle\}$ is a bisimulation. For that we define a function $\xi : R \to \mathcal{D}_\omega(R)$ as $\xi(\langle s_i, -\rangle)(\langle s_j, -\rangle) = d(s_i)(s_j)$ and we observe that the projection maps π_1 and π_2 are coalgebra homomorphisms, that is, the following diagram commutes.

$$\begin{array}{ccccc}
\langle s\rangle & \xleftarrow{\pi_1} & R & \xrightarrow{\pi_2} & \{A_i^n \mid s_i \in \langle s\rangle\} \\
{\scriptstyle d}\downarrow & (1) & \downarrow{\scriptstyle \xi} & (2) & \downarrow{\scriptstyle \delta_{\mathrm{Exp}_{\mathcal{D}_\omega}}} \\
\mathcal{D}_\omega(\langle s\rangle) & \xleftarrow[\mathcal{D}_\omega(\pi_1)]{} & \mathcal{D}_\omega(R) & \xrightarrow[\mathcal{D}_\omega(\pi_2)]{} & \mathcal{D}_\omega(\{A_i^n \mid s_i \in \langle s\rangle\})
\end{array}$$

$$\mathcal{D}_\omega(\pi_1)(\xi(\langle s_i, A_i^n\rangle))(s_j) = \sum_{\langle s_j,x\rangle\in R} \xi(\langle s_i, A_i^n\rangle)(\langle s_j, x\rangle) = d(s_i)(s_j) \tag{1}$$

$$\mathcal{D}_\omega(\pi_2)(\xi(\langle s_i, A_i^n\rangle))(A_j^n) = \sum_{\langle x,a_j^n\rangle\in R} \xi(\langle s_i, A_i^n\rangle)(\langle x, A_j^n\rangle) = d(s_i)(s_j) = \delta_{\mathrm{Exp}_{\mathcal{D}_\omega}}(A_i^n)(A_j^n) \tag{2}$$

For the second part of the theorem, We need to show that for every expression $\varepsilon \in \mathrm{Exp}_{\mathcal{D}_\omega}$ there is a finite $\mathcal{D}_\omega$-coalgebra (S, d) with $s \in S$ such that $s \sim_b \varepsilon$. We take $(S, d) = \langle\varepsilon\rangle$ and we observe that S is finite, because $S \subseteq cl(\varepsilon)$ (the proof of this inclusion is as in Theorem 4.9). □

The axiomatization of $\mathrm{Exp}_{\mathcal{D}_\omega}$ is a subset of the one for $\mathrm{Exp}_{\mathbb{R}^{\mathrm{Id}}}$, since some axioms, such as $\underline{\emptyset} \equiv 0 \cdot \varepsilon$, have no meaning in the restricted syntax. In this case the axiomatization for $\mathrm{Exp}_{\mathcal{D}_\omega}$ would contain the axioms for the fixed-point, plus (*Associativity*), (*Commutativity*) and $p \cdot \varepsilon \oplus p' \cdot \varepsilon = (p + p') \cdot \varepsilon$. The soundness of this axiomatization comes for free from the soundness in $\mathbb{R}^{\mathrm{Id}}$, because behavioural equivalence in $\mathbb{R}^{\mathrm{Id}}$ implies behavioural equivalence in $\mathcal{D}_\omega$. Completeness needs however a bit more of work: we need to prove that, for any two expressions in the new syntax, if they are provably equivalent using all

the axioms of $\mathrm{Exp}_{\mathbb{R}^{\mathrm{Id}}}$ then they must be provably equivalent using only the restricted set of axioms. Note that it is usually obvious which axioms one needs to keep for the restricted syntax.

For another example, consider the functors Id and $\mathcal{P}_\omega(\mathrm{Id})$. Let τ be the family of functions $\tau_S : S \to \mathcal{P}_\omega(S)$ mapping each $s \in S$ in the singleton set $\{s\}$. It is easy to see that τ is an injective natural transformation. With the above observation, we can also get regular expressions for the functor $\mathbb{M}^{\mathrm{Id}} \times \mathrm{Id}^A$ which, as discussed in Section 4, does not belong to our class of quantitative functors. By extending τ, we can construct an injective natural transformation $\mathbb{M}^{\mathrm{Id}} \times \mathrm{Id}^A \Rightarrow \mathbb{M}^{\mathrm{Id}} \times \mathcal{P}_\omega(\mathrm{Id})^A$.

In the same way, we can construct an injective natural transformation from the functor $\mathcal{D}_\omega(\mathrm{Id}) + (A \times \mathrm{Id}) + 1$ (which is the type of stratified systems, which we shall use as an example in the next section) into $\mathbb{R}^{\mathrm{Id}} + (A \times \mathcal{P}_\omega(\mathrm{Id})) + 1$. Since the latter is a quantitative functor, we can use its expressions and axiomatization for stratified systems. But since not all its expressions define stratified behaviours, we again will have to restrict the syntax.

6. Probabilistic systems

Many different types of probabilistic systems have been defined in the literature: examples include reactive, generative, stratified, alternating, (simple) Segala, bundle and Pnueli–Zuck. Each type corresponds to a functor, and the systems of a certain type are coalgebras of the corresponding functor. A systematic study of all these systems as coalgebras was made in [5]. In particular, Fig. 1 of [5] provides a full correspondence between types of systems and functors. By employing this correspondence and the results of the previous section, we can use our framework in order to derive regular expressions and axiomatizations for all these types of probabilistic systems.

In order to show the effectiveness of our approach, we have derived expressions and axioms for three different types of probabilistic systems: simple Segala, stratified, and Pnueli–Zuck.

Simple Segala systems Simple Segala systems are transition systems where both probability and non determinism are present. They are coalgebras of the functor $\mathcal{P}_\omega(\mathcal{D}_\omega(\mathrm{Id}))^A$. Each labelled transition leads, non-deterministically, to a probability distribution of states instead of a single state. An example is shown in Fig. 1(i).

We recall the expressions and axioms for simple Segala systems as shown in Table 1.

$$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \boxplus \varepsilon \mid \mu x.\varepsilon \mid x \mid a(\{\varepsilon'\}) \qquad \text{where } a \in A,\ p_i \in (0,1] \text{ and } \sum_{i \in 1\, ldots, n} p_i = 1$$

$$\varepsilon' ::= \bigoplus_{i \in 1,\ldots,n} p_i \cdot \varepsilon_i$$

$$(\varepsilon_1 \boxplus \varepsilon_2) \boxplus \varepsilon_3 \equiv \varepsilon_1 \boxplus (\varepsilon_2 \boxplus \varepsilon_3) \qquad \varepsilon_1 \boxplus \varepsilon_2 \equiv \varepsilon_2 \boxplus \varepsilon_1 \qquad \varepsilon \boxplus \underline{\emptyset} \equiv \varepsilon \qquad \varepsilon \boxplus \varepsilon \equiv \varepsilon$$

$$(\varepsilon'_1 \oplus \varepsilon'_2) \oplus \varepsilon'_3 \equiv \varepsilon'_1 \oplus (\varepsilon'_2 \oplus \varepsilon'_3) \qquad \varepsilon'_1 \oplus \varepsilon'_2 \equiv \varepsilon'_2 \oplus \varepsilon'_1 \qquad (p_1 \cdot \varepsilon) \oplus (p_2 \cdot \varepsilon) \equiv (p_1 + p_2) \cdot \varepsilon$$

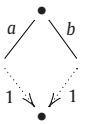
$$\varepsilon[\mu x.\varepsilon/x] \equiv \mu x.\varepsilon \qquad \gamma[\varepsilon/x] \equiv \varepsilon \Rightarrow \mu x.\gamma \equiv \varepsilon$$

Here, in order to avoid confusion, we use $\boxplus$ instead of $\oplus$ for the expressions at the top level, making a clear distinction between the idempotent (non-deterministic) and non-idempotent (probabilistic) sums. The syntax above was obtained from the canonically derived one by applying the restrictions arising from $\mathcal{D}_\omega$ and also some simplifications (syntactic sugar) which improve readability (in the spirit of what we showed before for $\mathrm{Exp}_{\mathcal{W}}$, the expressions for weighted automata).

As we showed in Section 5, to be completely formal we would have to prove Kleene's Theorem and the completeness of the axiomatization for the restricted syntax (as well as the correctness of the simplifications). The proofs would be based on the ones we showed for the functor $\mathcal{D}_\omega$ (and the ones for the simplifications similar to what we showed for $\mathrm{Exp}_{\mathcal{W}}$). We omit them here and show instead an example of application. The expression $a(\{1/2 \cdot \underline{\emptyset} \oplus 1/2 \cdot \underline{\emptyset}\}) \boxplus a(\{1/3 \cdot \underline{\emptyset} \oplus 2/3 \cdot \underline{\emptyset}\}) \boxplus b(\{1 \cdot \underline{\emptyset}\})$ is bisimilar to the top-most state in the simple Segala system depicted in Fig. 1(i). Using the axiomatization, we can derive:

$$\begin{aligned}
& a(\{1/2 \cdot \underline{\emptyset} \oplus 1/2 \cdot \underline{\emptyset}\}) \boxplus a(\{1/3 \cdot \underline{\emptyset} \oplus 2/3 \cdot \underline{\emptyset}\}) \boxplus b(\{1 \cdot \underline{\emptyset}\}) \\
\equiv\ & a(\{(1/2 + 1/2) \cdot \underline{\emptyset}\}) \boxplus a(\{(1/3 + 2/3) \cdot \underline{\emptyset}\}) \boxplus b(\{1 \cdot \underline{\emptyset}\}) \\
\equiv\ & a(\{1 \cdot \underline{\emptyset}\}) \boxplus a(\{1 \cdot \underline{\emptyset}\}) \boxplus b(\{1 \cdot \underline{\emptyset}\}) \\
\equiv\ & a(\{1 \cdot \underline{\emptyset}\}) \boxplus b(\{1 \cdot \underline{\emptyset}\})
\end{aligned}$$

Thus, we can conclude that the system presented in Fig. 1(i) is bisimilar to the following one:



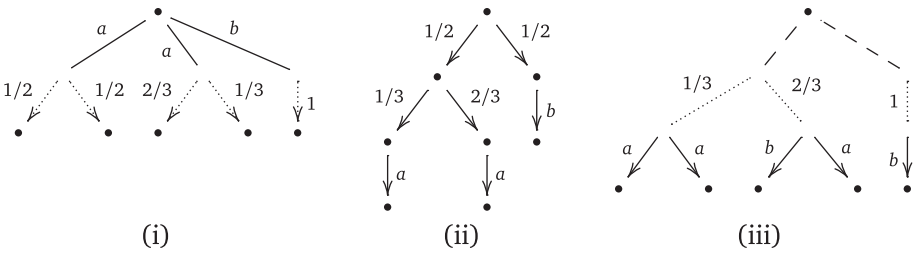


Fig. 1. (i) A simple Segala system, (ii) a stratified system and (iii) a Pnueli–Zuck system

The language and axiomatization we presented above are the same as the one presented in [17] (with the difference that in [17] a parallel composition operator was also considered). This is of course reassuring for the correctness of the general framework we presented. In the next two examples, we will present new results (that is syntax/axiomatizations which did not exist). This is where the generality starts paying off: not only one recovers known results but also derives new ones, all of this inside the same uniform framework.

Stratified systems. Stratified systems are coalgebras of the functor $\mathcal{D}_\omega(\mathrm{Id}) + (\mathrm{B} \times \mathrm{Id}) + 1$. Each state of these systems either performs unlabelled probabilistic transitions or one B-labelled transition or it terminates. We first derive expressions and axioms for $\mathbb{R}^{\mathrm{Id}} + (\mathrm{B} \times \mathcal{D}_\omega(\mathrm{Id})) + 1$ and then we restrict the syntax to characterize only $\mathcal{D}_\omega(\mathrm{Id}) + (\mathrm{B} \times \mathrm{Id}) + 1$-behaviours. This, together with the introduction of some syntactic sugar, leads to the following syntax and axioms.

$$\varepsilon ::= \mu x.\varepsilon \mid x \mid \langle b, \varepsilon \rangle \mid \bigoplus_{i \in 1,\ldots,n} p_i \cdot \varepsilon_i \mid \downarrow \qquad \text{where } b \in \mathrm{B},\ p_i \in (0,1] \text{ and } \sum_{i \in 1,\ldots,n} p_i = 1$$

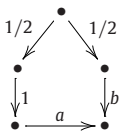
$$(\varepsilon_1 \oplus \varepsilon_2) \oplus \varepsilon_3 \equiv \varepsilon_1 \oplus (\varepsilon_2 \oplus \varepsilon_3) \qquad \varepsilon_1 \oplus \varepsilon_2 \equiv \varepsilon_2 \oplus \varepsilon_1 \qquad (p_1 \cdot \varepsilon) \oplus (p_2 \cdot \varepsilon) \equiv (p_1 + p_2) \cdot \varepsilon$$
$$\varepsilon[\mu x.\varepsilon/x] \equiv \mu x.\varepsilon \qquad \gamma[\varepsilon/x] \equiv \varepsilon \Rightarrow \mu x.\gamma \equiv \varepsilon$$

Here $\downarrow$, which denotes termination, corresponds to the canonically derived expression $r[r[1]]$, whilst $\langle b, \varepsilon \rangle$ corresponds to $r[l[l\langle b \rangle \oplus r\langle\{\varepsilon\}\rangle]]$.

We can use these axioms (together with Kleene's theorem) to reason about the system presented in Fig. 1(ii). The topmost state of this system is bisimilar to the expression

$$1/2 \cdot (1/3 \cdot \langle a, \downarrow \rangle \oplus 2/3 \cdot \langle a, \downarrow \rangle) \oplus 1/2 \cdot \langle b, \downarrow \rangle$$

which in turn is provably equivalent to $1/2 \cdot (1 \cdot \langle a, \downarrow \rangle) \oplus 1/2 \cdot \langle b, \downarrow \rangle$. That leads us to conclude that the aforementioned system is equivalent to the following simpler one.



The language of expressions we propose for these systems is a subset of the language originally proposed in [45] (there a parallel composition operator is also considered). More interestingly, there was no axiomatization of the language in [45] and thus the axiomatization we present here is completely new.

Pnueli–Zuck systems. These systems are coalgebras of the functor $\mathcal{P}_\omega\mathcal{D}_\omega(\mathcal{P}_\omega(\mathrm{Id})^A)$. Intuitively, the ingredient $\mathcal{P}_\omega(\mathrm{Id})^A$ denotes A-labelled transitions to other states. Then, $\mathcal{D}_\omega(\mathcal{P}_\omega(\mathrm{Id})^A)$ corresponds to a probability distribution of labelled transitions and then, each state of a $\mathcal{P}_\omega\mathcal{D}_\omega(\mathcal{P}_\omega(\mathrm{Id})^A)$-coalgebra performs a non-deterministic choice amongst probability distributions of labelled transitions. For an example, consider the system depicted in Fig. 1(iii).

We first derive expressions and axioms for the functor $\mathcal{P}_\omega(\mathbb{R}^{\mathcal{P}_\omega(\mathrm{Id})^A})$ and then we restrict the syntax to characterize only $\mathcal{P}_\omega\mathcal{D}_\omega(\mathcal{P}_\omega(\mathrm{Id})^A)$-behaviours (note that the existence of an injective natural transformation from $\mathcal{P}_\omega\mathcal{D}_\omega(\mathcal{P}_\omega(\mathrm{Id})^A)$ to $\mathcal{P}_\omega(\mathbb{R}^{\mathcal{P}_\omega(\mathrm{Id})^A})$ is a direct consequence of the existence of a natural transformation from $\mathcal{D}_\omega$ to $\mathbb{R}^{\mathrm{Id}}$ and the fact that $\mathcal{P}_\omega$ preserves monos). The expressions and axioms for these systems are the following:

$$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \boxplus \varepsilon \mid \mu x.\varepsilon \mid x \mid \{\varepsilon'\} \qquad \text{where } a \in A,\ p_i \in (0,1] \text{ and } \sum_{i \in 1,\ldots,n} p_i = 1$$
$$\varepsilon' ::= \bigoplus_{i \in 1,\ldots,n} p_i \cdot \varepsilon_i''$$
$$\varepsilon'' ::= \underline{\emptyset} \mid \varepsilon'' \boxplus \varepsilon'' \mid a(\{\varepsilon\})$$

$$(\varepsilon_1 \boxplus \varepsilon_2) \boxplus \varepsilon_3 \equiv \varepsilon_1 \boxplus (\varepsilon_2 \boxplus \varepsilon_3) \qquad \varepsilon_1 \boxplus \varepsilon_2 \equiv \varepsilon_2 \boxplus \varepsilon_1 \qquad \varepsilon \boxplus \underline{\emptyset} \equiv \varepsilon \qquad \varepsilon \boxplus \varepsilon \equiv \varepsilon$$
$$(\varepsilon_1' \oplus \varepsilon_2') \oplus \varepsilon_3' \equiv \varepsilon_1' \oplus (\varepsilon_2' \oplus \varepsilon_3') \qquad \varepsilon_1' \oplus \varepsilon_2' \equiv \varepsilon_2' \oplus \varepsilon_1' \qquad (p_1 \cdot \varepsilon'') \oplus (p_2 \cdot \varepsilon'') \equiv (p_1 + p_2) \cdot \varepsilon''$$
$$(\varepsilon_1'' \boxplus \varepsilon_2'') \boxplus \varepsilon_3'' \equiv \varepsilon_1'' \boxplus (\varepsilon_2'' \boxplus \varepsilon_3'') \quad \varepsilon_1'' \boxplus \varepsilon_2'' \equiv \varepsilon_2'' \boxplus \varepsilon_1'' \quad \varepsilon'' \boxplus \underline{\emptyset} \equiv \varepsilon'' \quad \varepsilon'' \boxplus \varepsilon'' \equiv \varepsilon''$$
$$\varepsilon[\mu x.\varepsilon/x] \equiv \mu x.\varepsilon \qquad \gamma[\varepsilon/x] \equiv \varepsilon \Rightarrow \mu x.\gamma \equiv \varepsilon$$

The expression $\{1/3 \cdot (a(\{\underline{\emptyset}\}) \boxplus a(\{\underline{\emptyset}\})) \oplus 2/3 \cdot (b(\{\underline{\emptyset}\}) \boxplus a(\{\underline{\emptyset}\}))\} \boxplus \{1 \cdot b(\{\underline{\emptyset}\})\}$ specifies the Pnueli–Zuck system in Fig. 1(iii). Note that we use the same symbol $\boxplus$ for denoting two different kinds of non-deterministic choice. This is safe, since they satisfy exactly the same axioms.

Both the syntax and the axioms we propose here for these systems are to the best of our knowledge new. In the past, these systems were studied using a temporal logic [33].

7. A slight variation on the monoidal exponentiation functor

In this section, we show a slight variation on the definition of the monoidal exponentiation functor which would allow for a cleaner derivation of syntax and axioms for certain functors, amongst which the probability functor.

In the spirit of [26], where this functor is defined, we shall call it constrained monoidal exponentiation functor.

Definition 7.1 (*Constrained monoidal exponentiation functor*)**.** Let $\mathbb{M}$ be a commutative monoid and $V \subseteq \mathbb{M}$ (V being the constraint). The constrained monoidal exponentiation functor $\mathbb{M}_V^- : \mathbf{Set} \to \mathbf{Set}$ is defined on sets as $\mathbb{M}_V^S = \{f \in \mathbb{M}_\omega^S \mid \sum_{s \in S} f(s) \in V\}$ and on functions as $\mathbb{M}_\omega^-$.

The probability functor $\mathcal{D}_\omega$ coincides now with $(\mathbb{R}_0^+)_{\{1\}}^-$ ($\mathbb{R}_0^+$ denotes the monoid of positive real numbers).

The expressions associated with this functor are the closed and guarded expressions given by the following BNF, where $x \in X$ and $m_i \in \mathbb{M}$,

$$\mathrm{Exp}_{\mathbb{M}_V^-} \ni \varepsilon ::= x \mid \mu x.\varepsilon \mid \bigoplus_{i \in I} m_i \cdot \varepsilon_i \text{ such that } \sum_{i \in I} m_i \in V$$

We note that instantiating this syntax for $(\mathbb{R}_0^+)_{\{1\}}^-$, one gets precisely the syntax we proposed in Eq. (13) for $\mathcal{D}_\omega$.

Providing a Kleene like theorem and a sound and complete axiomatization goes precisely as before (for the functor $\mathbb{M}_\omega^-$), with the minor difference that the axiom $0 \cdot \varepsilon \equiv \underline{\emptyset}$ has to be replaced by $0 \cdot \varepsilon \oplus m \cdot \varepsilon' \equiv m \cdot \varepsilon'$, since $\underline{\emptyset}$ is not a valid expression for this functor.

All of this seems to indicate that using the constrained monoidal exponentiation functor would have made it easier to define expressions and axiomatizations for quantitative systems. Although this would have been true for systems such as the simple Segala, it would not completely avoid the use of the technique we described in Section 5, which allows us to deal with a large class of functors: not only with $\mathcal{D}_\omega$ (embedded into $\mathbb{R}^{\mathrm{Id}}$) but also with mixed functors, such the one of stratified systems. Moreover, using this functor would require extra care when dealing with the modularity of the axiomatization, which we will illustrate next by means of an example. For these reasons, we decided to present the syntax and axiomatizations of all our running examples as a special instance of the general technique described in Section 5.

Example 7.2 (*Reactive probabilistic automata*)**.** Let us consider a very simple version of reactive probabilistic automata, coalgebras for the functor $\mathcal{D}_\omega(-)^A = ((\mathbb{R}_0^+)_{\{1\}}^{\mathrm{Id}})^A$.

The syntax modularly derived for this functor would be

$$\varepsilon ::= \underline{\emptyset} \mid \varepsilon \oplus \varepsilon \mid x \mid \mu x.\varepsilon \mid a(\varepsilon')$$
$$\varepsilon' ::= \bigoplus_{i \in I} p_i \cdot \varepsilon_i \qquad \text{such that } \sum_{i \in I} p_i = 1$$

In the axiomatization, we would (expect to) have the axiom $a(\varepsilon_1) \oplus a(\varepsilon_2) \equiv a(\varepsilon_1 \oplus \varepsilon_2)$. But now note how this could lead to an inconsistent specification, since $\varepsilon_1 \oplus \varepsilon_2$ will not be a valid expression anymore for $\mathcal{D}_\omega$ (if $\sum p_i = 1$ in both ε_1 and ε_2 then it will be 2 in $\varepsilon_1 \oplus \varepsilon_2$!).

In order to keep the axiomatization compositional we would have to require certain conditions on the set V in the functor $\mathbb{M}_V^-$. For instance, one of the possible conditions would be that V would have to be closed with respect to $+$, which would be a too strong condition to model $\mathcal{D}_\omega$.

8. Discussion

We presented a general framework to canonically derive expressions and axioms for quantitative regular behaviours. To illustrate the effectiveness and generality of our approach we derived expressions and equations for weighted automata,

simple Segala, stratified and Pnueli–Zuck systems. We recovered the syntaxes proposed in [13,17,45] for the first three models and the axiomatization of [17]. For weighted automata and stratified systems we derived new axiomatizations and for Pnueli–Zuck systems both a novel language of expressions and axioms. The process calculi in [13,17,45] also contained a parallel composition operator and thus they slightly differ from our languages that are more in the spirit of Kleene and Milner's expressions. In order to obtain a language, based on the one we defined in this chapter, which also includes other (user-defined) operators, such as parallel composition, we would like to study the connection with bialgebras and GSOS.

In [4,16,42] expressions without parallel composition are studied for probabilistic systems. These provide syntax and axioms for generative systems, Segala systems and alternating systems, respectively. For Segala systems our approach will derive the same language of [16], whilst the expressions in [42] differ from the ones resulting from our approach, since they use a probabilistic choice operator $+_p$. For alternating systems, our approach could bring some new insights, since in [4] only expressions without recursion are considered.

The derivation of the syntax and axioms associated with each quantitative functor is in the process of being implemented in the coinductive prover CIRC [30]. For the non-deterministic fragment everything can be done automatically, whereas for the functors described in Section 5, such as the probability functor, some user input is required, in order to define the syntactic restrictions. This will then allow for automatic reasoning about the equivalence of expressions specifying systems.

In this paper we studied coalgebras for **Set** functors. It is an interesting and challenging question to extend our results to other categories. In particular, it seems promising to study functors over metric spaces [43,44] or vector spaces [9,36].

Acknowledgments

The authors are grateful for useful comments from several people, amongst which Bartek Klin, Ana Sokolova, the anonymous reviewers of the conference version of this paper, the (very participative) audience of CONCUR'09 and the two anonymous reviewers of the present paper for their detailed reports which greatly improved the presentation.

References

[1] Luca Aceto, Zoltán Ésik, Anna Ingólfsdóttir, Equational axioms for probabilistic bisimilarity, in: Hélène Kirchner, Christophe Ringeissen (Eds.), AMAST, Lecture Notes in Computer Science, vol. 2422, Springer, 2002, pp. 239–253.
[2] Jos C.M. Baeten, Jan A. Bergstra, Scott A. Smolka, Axiomization probabilistic processes: Acp with generative probabililties (extended abstract), in: Cleaveland [14], pp. 472–485.
[3] Jos C.M. Baeten, Jan Willem Klop (Eds.), in: CONCUR '90, Theories of Concurrency: Unification and Extension, Amsterdam, The Netherlands, August 27–30, 1990, Proceedings, Lecture Notes in Computer Science, vol. 458, Springer, 1990.
[4] Emanuele Bandini, Roberto Segala, Axiomatizations for probabilistic bisimulation, in: Fernando Orejas, Paul G. Spirakis, Jan van Leeuwen (Eds.), ICALP, Lecture Notes in Computer Science, vol. 2076, Springer, 2001, pp. 370–381.
[5] Falk Bartels, Ana Sokolova, Erik P. de Vink, A hierarchy of probabilistic system types, Theor. Comput. Sci. 327 (1–2) (2004) 3–22.
[6] Filippo Bonchi, Marcello M. Bonsangue, Jan J.M.M. Rutten, Alexandra Silva, Deriving syntax and axioms for quantitative regular behaviours, in: Bravetti, Zavattaro [10], pp. 146–162.
[7] Marcello M. Bonsangue, Jan J.M.M. Rutten, Alexandra Silva, Coalgebraic logic and synthesis of Mealy machines, in: Roberto M. Amadio (Ed.), FoSSaCS, Lecture Notes in Computer Science, vol. 4962, Springer, 2008, pp. 231–245.
[8] Marcello M. Bonsangue, Jan J.M.M. Rutten, Alexandra Silva, An algebra for Kripke polynomial coalgebras, in: LICS, IEEE Computer Society, 2009, pp. 49–58.
[9] Michele Boreale, Weighted bisimulation in linear algebraic form, in: Bravetti, Zavattaro [10], pp. 163–177.
[10] Mario Bravetti, Gianluigi Zavattaro (Eds.), in: CONCUR 2009 – Concurrency Theory, 20th International Conference, CONCUR 2009, Bologna, Italy, September 1–4, 2009, Proceedings, Lecture Notes in Computer Science, vol. 5710, Springer, 2009.
[11] Janusz A. Brzozowski, Derivatives of regular expressions, J. ACM 11 (4) (1964) 481–494.
[12] Peter Buchholz, Bisimulation relations for weighted automata, Theor. Comput. Sci. 393 (1–3) (2008) 109–123.
[13] Peter Buchholz, Peter Kemper, Quantifying the dynamic behavior of process algebras, in: Luca de Alfaro, Stephen Gilmore (Eds.), PAPM-PROBMIV, Lecture Notes in Computer Science, vol. 2165, Springer, 2001, pp. 184–199.
[14] Rance Cleaveland (Ed.), in: CONCUR '92, Third International Conference on Concurrency Theory, Stony Brook, NY, USA, August 24–27, 1992, Proceedings, Lecture Notes in Computer Science, vol. 630, Springer, 1992.
[15] Pedro R. D'Argenio, Holger Hermanns, Joost-Pieter Katoen, On generative parallel composition, Electr. Notes Theor. Comput. Sci. 22 (1999) 30–54.
[16] Yuxin Deng, Catuscia Palamidessi, Axiomatizations for probabilistic finite-state behaviors, in: Vladimiro Sassone (Ed.), FoSSaCS, Lecture Notes in Computer Science, vol. 3441, Springer, 2005, pp. 110–124.
[17] Yuxin Deng, Catuscia Palamidessi, Jun Pang, Compositional reasoning for probabilistic finite-state behaviors, in: Aart Middeldorp, Vincent van Oostrom, Femke van Raamsdonk, Roel C. de Vrijer (Eds.), Processes, Terms and Cycles, Lecture Notes in Computer Science, vol. 3838, Springer, 2005, pp. 309–337.
[18] Manfred Droste, Paul Gastin, Weighted automata and weighted logics, in: Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, Moti Yung (Eds.), ICALP, Lecture Notes in Computer Science, vol. 3580, Springer, 2005, pp. 513–525.
[19] Alessandro Giacalone, Chi-Chang Jou, Scott A. Smolka, Algebraic reasoning for probabilistic concurrent systems, in: M. Broy, C.B. Jones (Eds.), Proceedings of Working Conference on Programming Concepts and Methods, IFIP TC 2, 1990.
[20] H. Peter Gumm, Tobias Schröder, Monoid-labeled transition systems, Electr. Notes Theor. Comput. Sci. 44 (1) (2001) 185–204.
[21] H. Peter Gumm, Tobias Schröder, Products of coalgebras, Algebra Universalis 46 (2001) 163–185.
[22] H. Peter Gumm, Tobias Schröder, Coalgebras of bounded type, Math. Struct. Comput. Sci. 12 (5) (2002) 565–578.
[23] Hans Hansson, Bengt Jonsson, A logic for reasoning about time and reliability, Formal Asp. Comput. 6 (5) (1994) 512–535.
[24] Chi-Chang Jou, Scott A. Smolka, Equivalences, congruences, and complete axiomatizations for probabilistic processes, in: Baeten, Klop [3], pp. 367–383.
[25] Stephen Kleene, Representation of events in nerve nets and finite automata, Autom. Stud. (1956) 3–42.
[26] Bartek Klin, Structural operational semantics for weighted transition systems, in: Jens Palsberg (Ed.), Semantics and Algebraic Specification, Lecture Notes in Computer Science, vol. 5700, Springer, 2009, pp. 121–139.
[27] Dexter Kozen, A completeness theorem for Kleene algebras and the algebra of regular events, in: Proceedings, Sixth Annual IEEE Symposium on Logic in Computer Science, 15–18 July, LICS, IEEE Computer Society, Amsterdam, The Netherlands, 1991, pp. 214–225.
[28] Kim Guldstrand Larsen, Arne Skou, Bisimulation through probabilistic testing, Inform. Comput 94 (1) (1991) 1–28.
[29] Kim Guldstrand Larsen, Arne Skou, Compositional verification of probabilistic processes, in: Cleaveland [14], pp. 456–471.

[30] Dorel Lucanu, Eugen-Ioan Goriac, Georgiana Caltais, Grigore Rosu, Circ: a behavioral verification tool based on circular coinduction, in: Alexander Kurz, Marina Lenisa, Andrzej Tarlecki (Eds.), CALCO, Lecture Notes in Computer Science, vol. 5728, Springer, 2009, pp. 433–442.
[31] Robin Milner, A complete inference system for a class of regular behaviours, J. Comput. Syst. Sci. 28 (3) (1984) 439–466.
[32] Michael W. Mislove, Joël Ouaknine, James Worrell, Axioms for probability and nondeterminism, Electr. Notes Theor. Comput. Sci. 96 (2004) 7–28.
[33] Amir Pnueli, Lenore D. Zuck, Probabilistic verification by tableaux, in: LICS, IEEE Computer Society (1986) 322–331.
[34] Michael O. Rabin, Probabilistic automata, Inform. Control 6 (3) (1963) 230–245.
[35] Jan J.M.M. Rutten, Universal coalgebra: a theory of systems, Theor. Comput. Sci. 249 (1) (2000) 3–80.
[36] Jan J.M.M. Rutten, Coalgebraic foundations of linear systems, in: Till Mossakowski, Ugo Montanari, Magne Haveraaen (Eds.), CALCO, Lecture Notes in Computer Science, vol. 4624, Springer, 2007, pp. 425–446.
[37] Arto Salomaa, Two complete axiom systems for the algebra of regular events, J. ACM 13 (1) (1966) 158–169.
[38] Marcel Paul Schützenberger, On the definition of a family of automata, Inform. Control 4 (2–3) (1961) 245–270.
[39] Roberto Segala, Modeling and Verification of Randomized Distributed Real-time Systems, Ph.D. thesis, MIT, Department of EECS, 1995.
[40] Roberto Segala, Nancy A. Lynch, Probabilistic simulations for probabilistic processes, in: Bengt Jonsson, Joachim Parrow (Eds.), CONCUR, Lecture Notes in Computer Science, vol. 836, Springer, 1994, pp. 481–496.
[41] Scott A. Smolka, Bernhard Steffen, Priority as extremal probability, in: Baeten, Klop [3], pp. 456–466.
[42] Eugene W. Stark, Scott A. Smolka, A complete axiom system for finite-state probabilistic processes, in: Gordon D. Plotkin, Colin Stirling, Mads Tofte (Eds.), Proof, Language, and Interaction, The MIT Press, 2000, pp. 571–596.
[43] Daniele Turi, Jan J.M.M. Rutten, On the foundations of final coalgebra semantics: non-well-founded sets, partial orders, metric spaces, Math. Struct. Comput. Sci. 8 (5) (1998) 481–540.
[44] Franck van Breugel, James Worrell, Approximating and computing behavioural distances in probabilistic transition systems, Theor. Comput. Sci. 360 (1–3) (2006) 373–385.
[45] Rob J. van Glabbeek, Scott A. Smolka, Bernhard Steffen, Reactive, generative and stratified models of probabilistic processes, Inform. Comput. 121 (1) (1995) 59–80.
[46] Moshe Y. Vardi, Automatic verification of probabilistic concurrent finite-state programs, in: FOCS, Springer (1985) 327–338.