Journal of Logical and Algebraic Methods in Programming 94 (2018) 184–199

Contents lists available at ScienceDirect

Journal of Logical and Algebraic Methods in Programming

www.elsevier.com/locate/jlamp

# A complete logic for behavioural equivalence in coalgebras of finitary set functors

David Sprunger

*Department of Mathematics, Indiana University, 831 E. 3rd Street, Bloomington, IN 47401, USA*

A R T I C L E I N F O

*Article history:*
Received 14 August 2016
Received in revised form 21 January 2017
Accepted 5 May 2017
Available online 9 May 2017

*Keywords:*
Behavioural equivalence
Finitary functor
Bisimulation up to
Signature
Compositional presentation

A B S T R A C T

This paper presents a sound and complete sequent-style deduction system for determining behavioural equivalence in coalgebras of finitary set functors preserving weak pullbacks. We also prove soundness without the weak pullback requirement. Finitary set functors are investigated because they are quotients of polynomial functors: the polynomial functor provides a ready-made signature and the quotient provides necessary additional axioms. We also show that certain operations on functors can be expressed with uniform changes to the presentations of the input functors, making this system compositional for a range of widely-studied classes of functors, including the Kripke polynomial functors. Our system has roots in the $FLR_0$ proof system of Moschovakis et al., particularly as used by Moss, Wennstrom, and Whitney for non-wellfounded sets. Similarities can also be drawn to expression calculi in the style of Bonsangue, Rutten, and Silva.



## 1. Introduction

In this paper, we propose a logic for detecting behaviourally equivalent states in coalgebras of finitary Set-endofunctors. Finitary functors are special because they have presentations whereby they can be represented as the quotient of a signature functor by a collection of equations.[1] The signature provides a syntax in which coalgebras can be expressed, and the equations add the axioms necessary to distinguish reasoning among functors with similar signatures.

In particular, we will consider *specifications* on sets of variables in signatures of finitary functors. These are total assignments of variables to terms which serve as definitions, and may be considered a recasting of the long history of systems of simultaneous equations going back to Kahn, Manna and Vuillemin, and Lawvere. These specifications are also coalgebras for the signature functor, and the quotient relationship between a finitary functor and its signature functor creates a correspondence between coalgebras of the finitary functor and these specifications.

Our system is comparable to $FLR_0$, as considered by Moss in [2] and Moss et al. in [3]. $FLR_0$ has distinctive terms of the form $A_0$ **where** $\{x_1 = A_1, \dots, x_n = A_n\}$. This **where** operator binds variables which may occur in $A_0$, while also restricting their interpretation. As a consequence, many of the $FLR_0$ rules concern moving a definition in and out of a subscope or evolving the term before the **where** clause. We avoid these issues, roughly, by fixing a specification sending $x_i$ to $A_i$ and considering terms with this context backgrounded. Additionally, $FLR_0$ and the full $FLR$ language are intended as general languages of recursion with semantics of various flavors. The application here to coalgebras distinguishes our version somewhat.

*E-mail address:* dasprung@indiana.edu.

[1] Finitary functors in finitely presentable categories outside of Set may also have finitary presentations, see [1].

http://dx.doi.org/10.1016/j.jlamp.2017.05.001
2352-2208/

We might also compare this work to that of Bonsangue et al. in [4] and [5] or Milius' related work in the setting of vector spaces [6], where a $\mu$ operator provides a similar variable binding. The work of Bonsangue et al. feature an inductive class of functors, the Kripke polynomial functors, and a syntax of expressions based on that inductive class. They build a sound and complete axiomatization for these expressions which is compositional, meaning the expressions and laws involved are built in parallel with the definition of the functor. We show that the presentations involved in our setting enjoy similar compositional properties.

In Chapter 5 of Silva's PhD thesis [7] and the related paper [8], she gives an extension of this $\mu$ calculus to finitary functors, demonstrating that the expressions of this calculus exactly coincide with the behaviours of locally finite coalgebras. However, at the end of this work, questions regarding axiomatization and uniform proofs of soundness and completeness for the system are left open.

We are able to prove soundness and completeness for our logic for the finitary functors which preserve weak pullbacks, a common condition with numerous pleasant coalgebraic consequences including that bisimilarity and behavioural equivalence coincide, see Rutten [9]. In particular, polynomial functors and the finite powerset functor preserve weak pullbacks, so the functors in our setting properly include those of Bonsangue et al. and Moss et al.

This paper is a revised and extended version of the conference paper [10]. This revision includes a new section (Section 6) with some new results for functors which do not preserve weak pullbacks. In particular, we show the proposed logic is still sound for specifications for functors which do not preserve weak pullbacks.

*Outline.* In Section 2, we briefly recall some background on coalgebras, signatures, and finitary functors. This section introduces the interplay between coalgebras of a finitary functor and coalgebras of its related signature functor that are of central importance to later sections. In Section 3, we introduce bisimulation up-to-presentation, a novel up-to technique which permits expressing bisimulations for finitary functors in terms of bisimulations for their signatures. In Section 4, we present a formal proof system capturing the notion of bisimulation up-to-presentation. We show this system is sound and complete in the sense that it detects the bisimilarity of states in coalgebras for finitary functors preserving weak pullbacks. In Section 5, we note that signatures for previously studied inductive classes of these functors—including the Kripke polynomial functors—can be constructed compositionally. This allows the proof system developed in Section 4 to be constructed compositionally as well. In Section 6, we show the proof system developed in Section 4 is sound even for finitary functors which do not preserve weak pullbacks.

## 2. Background

In this section, we recall definitions and basic results about coalgebras, finitary signatures, finitary functors, and introduce the notion of a specification. Our setting is the category Set, and all functors are assumed to be Set-endofunctors. Additionally, we will often assume functors preserve weak pullbacks, but make a special note when this assumption is needed.

### 2.1. Coalgebras

Given a Set-endofunctor $F$, an $F$-*coalgebra* is a set $X$ together with a map $f : X \to FX$. The set is often called the *carrier* of the coalgebra, while $f$ gives its *structure* or *dynamics*.

A *coalgebra morphism* from an $F$-coalgebra $(X, f)$ to another $F$-coalgebra $(Y, g)$ is a map $\varphi : X \to Y$ such that the following diagram commutes:

$$\begin{array}{ccc} X & \xrightarrow{f} & FX \\ {\scriptstyle\varphi}\downarrow & & \downarrow{\scriptstyle F\varphi} \\ Y & \xrightarrow{g} & FY \end{array}$$

$F$-coalgebras together with coalgebra morphisms between them form a category, denoted $\mathbf{Coalg}_F$. Of particular interest are the $F$ where $\mathbf{Coalg}_F$ has a final object. This final coalgebra has a natural interpretation as a semantic object: the unique coalgebra morphism from any $F$-coalgebra into the final one can be considered a semantic map. Therefore, points in coalgebras which have the same image in the final coalgebra can be considered semantically or behaviourally equivalent to one another.

A related notion is that of an (Aczel–Mendler) $F$-*bisimulation* on a coalgebra. An $F$-bisimulation is a relation $R \subseteq X \times X$ such that there is an $F$-coalgebra structure on $R$, $\rho$, such that the following diagram commutes[2]:

[2] Throughout this paper we will write $\pi_i$ for the more cumbersome $\pi_i|_R$.

$$\begin{array}{ccccc} X & \xleftarrow{\pi_1} & R & \xrightarrow{\pi_2} & X \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \rho} & & \downarrow{\scriptstyle f} \\ FX & \xleftarrow[F\pi_1]{} & FR & \xrightarrow[F\pi_2]{} & FX \end{array}$$

Roughly speaking, a relation is a bisimulation if two points in a coalgebra being related implies their structures are also related. This gives a different notion of equivalence, which is known to coincide with the behavioural equivalence for weak pullback preserving functors. For more details, we refer the reader to Rutten [9].

### 2.2. Finitary signatures and functors

A *finitary signature* is a set $\Sigma$ with a map $ar : \Sigma \to \omega$. In the sequel, we often abbreviate "finitary signature" to "signature" and refer to the signature as $\Sigma$ instead of $(\Sigma, ar)$. The elements of $\Sigma$ are the *symbols* of the signature, and each symbol $f \in \Sigma$ has the *arity* $ar(f)$. The collection of all symbols with arity $n$ is denoted $\Sigma_n$.

Each finitary signature has an associated *signature functor*, $H_\Sigma$, given by $\coprod_n \Sigma_n \times X^n$. We denote a typical element of $H_\Sigma X$ by $f(x_1, \dots, x_{ar(f)})$ if $x_i \in X$ or $f(\vec{x})$ if $\vec{x} : ar(f) \to X$. $H_\Sigma X$ is often referred to as the set of all "flat terms" using symbols from $\Sigma$ with variables from $X$.

$F$ is a *finitary functor* if there is a finitary signature $\Sigma$ together with a (pointwise) epic natural transformation $\epsilon : H_\Sigma \twoheadrightarrow F$.[3] If $F$ is a finitary functor, we say $(\Sigma, \epsilon)$ is a *presentation* of $F$. Finitary functors have a number of alternate characterizations, including functors which preserve $\omega$-filtered colimits [1].

**Example 1.** For each set $A$, the constant functor $FX = A$ is finitary with signature $\Sigma = \Sigma_0 = A$ and the transformation $\epsilon$ with components $\epsilon_X : a() \mapsto a$.

**Example 2.** The identity functor is finitary with signature $\Sigma = \Sigma_1 = \{*\}$ and the transformation $\epsilon$ with components $\epsilon_X : *(x) \mapsto x$.

**Example 3.** The finite powerset functor $\mathcal{P}_\omega$ is finitary with $\Sigma_n = \{\sigma_n\}$ and the transformation $\epsilon_X : \sigma_n(x_1, \dots, x_n) \mapsto \{x_1, \dots, x_n\}$. Note that unlike the previous two transformations, this $\epsilon$ is not an isomorphism.

**Example 4.** The 3 powerset functor $\mathcal{P}_3$, which assigns each set to the set of its subsets of cardinality $< 3$, is finitary with signature $\Sigma_n = \{\sigma_n\}$ for $0 \le n < 3$ and the same $\epsilon$ as in $P_\omega$, restricted to the smaller set of terms.

**Example 5.** The functor $ZX = \{0, 1\} \times X \times X$ is finitary with signature $\Sigma = \Sigma_2 = \{0\text{:zip}, 1\text{:zip}\}$ and transformation given by

$$\epsilon_X(i\text{:zip}(x_1, x_2)) = (i, x_1, x_2).$$

We call this the "zip functor" in the sequel, though this is not standard terminology. We should also note that the symbol ':' should be thought of as a `cons` operator here, not as a typing operator.

**Example 6.** Let $\mathbb{K}$ be a semiring. The valuation functor $\mathbb{K}_\omega^{Id}$ is less well-known, so we describe it briefly. The functor $\mathbb{K}_\omega^{Id}$ sends a set $X$ to the set of finitely supported functions from $X$ to $\mathbb{K}$, that is $\mathbb{K}_\omega^X = \{f \in \mathbb{K}^X : X \setminus f^{-1}(0) \text{ is finite}\}$. On functions, the functor works by direct images: if $h : X \to Y$, then $\mathbb{K}_\omega^h(f)(y) = \sum_{x:h(x)=y} f(x)$. We should also note that the finite powerset functor mentioned in Example 3 above is a special case of this functor with $\mathbb{K} = \mathbb{B}$, the Boolean semiring.

In any case, the functor $\mathbb{K}_\omega^{Id}$ is finitary with a signature $\Sigma_n = \mathbb{K}^n$ and transformation given by $\epsilon_X : (l_1, \dots, l_n)(x_1, \dots, x_n) \mapsto \lambda x. \sum_{i:x_i=x} l_i$ for $n \ge 1$ and $\epsilon_X : () \mapsto \lambda x.0$ for $n = 0$.

**Example 7.** The Aczel–Mendler functor $(-)_2^3$, appearing in [12], is finitary with signature $\Sigma = \Sigma_2 = \{\sigma, \tau, \rho\}$ and transformation

$$\epsilon_X : \sigma(x, y) \mapsto (x, x, y) \qquad \epsilon_X : \tau(x, y) \mapsto (y, x, x) \qquad \epsilon_X : \rho(x, y) \mapsto (x, y, x)$$

These first five examples preserve weak pullbacks, while the last two may not,[4] so we will discuss them in greater detail in Section 6.

[3] That is, we are assuming each component $\epsilon_X$ is epic. For natural transformations between functors into Set, pointwise epic and epic in the functor category coincide [11, p. 91].

[4] The Aczel–Mendler functor certainly does not, as was shown at its introduction in [12]. Whether the valuation functor preserves weak pullbacks depends on properties of the underlying semiring $\mathbb{K}$. $\mathbb{K}_\omega^{Id}$ does not preserve weak pullbacks if and only if there are elements of $\mathbb{K}$ (except 0) which sum to 0. Therefore, if $\mathbb{K}$ is a ring, this functor does not preserve weak pullbacks. (In the Boolean semiring case $\mathbb{K} = \mathbb{B}$, the only set which joins to 0 is $\{0\}$ so $\mathcal{P}_\omega$ does preserve weak pullbacks.) For more details, we refer the reader to [13].

A *specification in the signature* $\Sigma$ is an $H_\Sigma$-coalgebra, a set $X$ together with a function $d : X \to H_\Sigma X$. Elements of $X$ are called *variables*, and $d$ gives their *definition*. Every specification in the signature of a finitary functor $F$ gives rise to an $F$-coalgebra: given $d : X \to H_\Sigma X$ postcomposing with $\epsilon_X$ yields $\epsilon_X \circ d : X \to FX$.

A single $F$-coalgebra $(X, f)$ will correspond to potentially many specifications in its signature. For each section $s$ of $\epsilon_X$, the composition $s \circ f : X \to H_\Sigma X$ is a specification in $\Sigma$. The $F$-coalgebra related to each of these specifications will be, not surprisingly, $(X, f)$. Note at least one section of $\epsilon_X$ is guaranteed to exist since Set has split epis.

At a broad level, this paper could be seen as an attempt to use the quotient relationship $\epsilon : H_\Sigma \twoheadrightarrow F$ between the functors $H_\Sigma$ and $F$ to understand the relationships between $H_\Sigma$- and $F$-coalgebras and particularly the relationships between $H_\Sigma$- and $F$-bisimulations. We hereafter reserve "specification" to mean an $H_\Sigma$-coalgebra, and the undecorated "coalgebra" to mean $F$-coalgebra.

Since we can readily recast specifications and coalgebras for finitary Set functors, we translate standard notions from coalgebras to specifications. For example, $R$ is an *$F$-bisimulation* on the specification $(X, d)$ when it is an $F$-bisimulation on the coalgebra $(X, \epsilon_X \circ d)$, the *standard semantics* for a variable in a specification is its image in a given final $F$-coalgebra, and two variables are *behaviourally equivalent* when they have the same standard semantics. Note that though the standard semantics of a variable depends on a choice (of isomorphic versions) of the final coalgebra, whether two variables are behaviourally equivalent is independent of this choice.

We write $(x, y) \in \vDash_{(X,d)}$ or more commonly $\vDash_{(X,d)} x = y$ when $x$ and $y$ are behaviourally equivalent states in the specification $(X, d)$. When the specification is clear from context, we write $(x, y) \in \vDash$ or $\vDash x = y$.

**Example 8.** We can give a specification for the zip functor with $X = \{x, y, z, w\}$ and

$$\begin{aligned} d(x) &= 0{:}\mathrm{zip}(y, z) & d(y) &= 1{:}\mathrm{zip}(x, w) \\ d(z) &= 1{:}\mathrm{zip}(z, w) & d(w) &= 0{:}\mathrm{zip}(w, z) \end{aligned}$$

As shown by Kupke and Rutten in [14] and Grabmayer et al. in [15], a final coalgebra for this functor is the set of streams in $\{0, 1\}$.[5] With this final coalgebra in mind the standard semantics for $x$ is the Thue–Morse sequence.

**Example 9.** Another zip specification, for $Y = \{x, y, z, w, u, v, q\}$ is given by

$$\begin{aligned} d(x) &= 0{:}\mathrm{zip}(y, z) & d(y) &= 1{:}\mathrm{zip}(x, v) & d(v) &= 0{:}\mathrm{zip}(w, u) \\ d(w) &= 0{:}\mathrm{zip}(w, z) & d(z) &= 1{:}\mathrm{zip}(z, w) & d(u) &= 1{:}\mathrm{zip}(u, v) \\ & & d(q) &= 0{:}\mathrm{zip}(y, u) & & \end{aligned}$$

In this specification, the states $x$ and $q$ are behaviourally equivalent. Our goal is to give a uniform account for detecting this behavioural equivalence.

Note also $x$ in this example and $x$ in Example 8 are behaviourally equivalent. We could consider the problem of showing the equivalence of two variables in two separate specifications, but by taking the disjoint union of the two specifications and determining equivalence within this single joint specification we get the same effect.

## 3. Bisimulation up to presentation

In this section, we introduce the notion of bisimulation up to presentation. Roughly, bisimulations up to presentation are $H_\Sigma$-bisimulations relaxed up to the kernel of $\epsilon$ in such a way that they correspond nicely to $F$-bisimulations. This allows us to detect $F$-bisimulations using $H_\Sigma$-bisimulations and so-called $\epsilon$ laws. We also give an alternate characterization of bisimulation up to presentation and several related sufficient criteria to conclude that a relation is a subset of the bisimilarity relation.

Since bisimulation up to presentation provides an alternate criterion which suffices for detecting bisimulations, we have intentionally named this type of relation in the style of enhanced coalgebraic bisimulations studied recently by Rot et al. [16] with veins of research going back to Milner, Park, Sangiorgi, and others. We are also struck by the similarities between the results in Section 3.2 and the flavor of standard up-to results. It is possible that a fibrational approach will yield some connections between these bodies of work, though there is not currently a compelling account of this connection.

Recall the standard (Aczel–Mendler) bisimulation diagram for an $H_\Sigma$-bisimulation on $X$:

$$\begin{tikzcd} X \arrow[d, "d"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\rho"] & X \arrow[d, "d"] \\ H_\Sigma X & H_\Sigma R \arrow[l, "H_\Sigma \pi_1"] \arrow[r, "H_\Sigma \pi_2"'] & H_\Sigma X \end{tikzcd}$$

[5] Kupke and Rutten actually considered a certain subcategory of $\mathbf{Coalg}_Z$, and showed the set of streams—with a slightly different structure than considered by Grabmayer—is final in this subcategory.

We say $R \subseteq X \times X$ is a *bisimulation up to the presentation* $(\Sigma, \epsilon)$ if there is a $\rho : R \to H_\Sigma R$ such that $\epsilon_X \circ d \circ \pi_i = \epsilon_X \circ H_\Sigma \pi_i \circ \rho$ for $i \in \{1, 2\}$. That is, $\rho$ does not quite have to make $R$ an $H_\Sigma$-coalgebra bisimulation: the paths in the diagram above may become equal after applying $\epsilon_X$ instead of commuting outright.

$$\begin{tikzcd} R \arrow[r, "\pi_i"] \arrow[d, "\rho"'] & X \arrow[r, "d"] & H_\Sigma X \arrow[d, "\epsilon_X"] \\ H_\Sigma R \arrow[r, "H_\Sigma \pi_i"'] & H_\Sigma X \arrow[r, "\epsilon_X"'] & FX \end{tikzcd}$$

**Theorem 1.** *For all specifications $(X, d)$ for a finitary set functor with presentation $(\Sigma, \epsilon)$, a relation $R \subseteq X \times X$ is an $F$-bisimulation if and only if it is a bisimulation up to the presentation $(\Sigma, \epsilon)$.*

**Proof.** $(\Leftarrow)$ Let $\rho$ give $R$ the structure of a bisimulation up to the presentation $(\Sigma, \epsilon)$. Then $\epsilon_R \circ \rho$ gives $R$ an $F$-coalgebra structure such that

$$\begin{tikzcd} X \arrow[d, "\epsilon_X \circ d"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\epsilon_R \circ \rho"] & X \arrow[d, "\epsilon_X \circ d"] \\ FX & FR \arrow[l, "F\pi_1"] \arrow[r, "F\pi_2"'] & FX \end{tikzcd}$$

commutes, so $R$ is an $F$-bisimulation.

$(\Rightarrow)$ Given an $F$-bisimulation structure $\phi$ on $R$, we claim $s \circ \phi$ gives a bisimulation up to presentation structure to $R$ where $s$ is any section of $\epsilon_R$. To see this, note $\epsilon_X \circ H_\Sigma \pi_i \circ s \circ \phi = F\pi_i \circ \epsilon_R \circ s \circ \phi = F\pi_i \circ \phi = \epsilon_X \circ d \circ \pi_i$ $\square$

We note that this statement is not that the $F$-bisimulation structures are in 1–1 correspondence with the bisimulation up to presentation structures, but that the carriers are in a 1–1 correspondence. Much like the correspondence between coalgebras and specifications, there is a possibly distinct bisimulation up to presentation structure for each section of $\epsilon_R$.

**Corollary 1.** *The biggest $F$-bisimulation on a coalgebra is the biggest bisimulation up to the presentation $(\Sigma, \epsilon)$ on that coalgebra.*

We denote bisimilarity, the biggest $F$-bisimulation on a coalgebra, by $\sim$. The bisimilarity relation is known to be an equivalence relation on all coalgebras for all Set functors preserving weak pullbacks, see e.g. Rutten [9].

### 3.1. An explicit characterization

We defined a bisimulation up to presentation using a variation of the Aczel–Mendler diagram, but there is a more concrete characterization for bisimulations up to presentation which we describe in this section.

For flat terms $\alpha, \beta \in H_\Sigma X$ we write $\alpha =_\epsilon \beta$ to mean $\epsilon_X(\alpha) = \epsilon_X(\beta)$. If $\alpha =_\epsilon \beta$ we say this is an $\epsilon$ *law*, or that $\alpha$ may be rewritten to $\beta$ using $\epsilon$ laws. Note the $=_\epsilon$ relation on $H_\Sigma X$ is an equivalence relation.

**Definition 1** *($c(R)$).* The *flat contextual closure* of a relation $R \subseteq X \times X$ is the relation $c(R) \subseteq H_\Sigma X \times H_\Sigma X$ defined by $f(x_1, \ldots x_{ar(f)})\ c(R)\ f(y_1, \ldots, y_{ar(f)})$ if and only if $x_i\ R\ y_i$ for all $1 \le i \le ar(f)$.

We denote the pointwise composition of relations $R$ and $S$ by $R \bullet S$. That is, $x(R \bullet S)z$ if and only if there exists a $y$ such that $xRy$ and $ySz$.

**Definition 2** *($\sim_R$).* Given a relation $R$ on $X$, we define $\sim_R$ to be $=_\epsilon \bullet\ c(R) \bullet =_\epsilon$, a relation on $H_\Sigma X$.

Since $\sim_R$ also depends on the transformation $\epsilon$ it would be more proper to denote it $\sim_{R,\epsilon}$, but since $\epsilon$ is standard for each functor we elide it from the notation.

Here we also emphasize the distinction between two very similar symbols: $\sim$ denotes bisimilarity on $X$, and has no direct relationship with the symbol $\sim_R$ just defined.

**Theorem 2.** *Given a finitary functor $F$ and a specification $(X, d)$ in that functor's signature, $R \subseteq X \times X$ is a bisimulation up to the presentation $(\Sigma, \epsilon)$ if and only if $xRy$ implies $d(x) \sim_R d(y)$. More explicitly, for each $(x, y) \in R$, there is an $f \in \Sigma_n$ and $(x_1, y_1), \ldots, (x_n, y_n) \in R$ such that:*

1. $d(x) =_\epsilon f(x_1, \ldots, x_n)$
2. $d(y) =_\epsilon f(y_1, \ldots, y_n)$

**Proof.** ($\Rightarrow$) Suppose we have $\rho : R \to H_\Sigma R$ such that $\epsilon_X \circ d \circ \pi_i = \epsilon_X \circ H_\Sigma \pi_i \circ \rho$. Let $(x, y) \in R$ and write $\rho(x, y) = f((x_1, y_1), \dots, (x_n, y_n))$ where $f \in \Sigma$ and $(x_i, y_i) \in R$. Then $(H_\Sigma \pi_1 \circ \rho)(x, y) = f(x_1, \dots, x_n)$, so by the hypothesis on $\rho$, $d(x) =_\epsilon f(x_1, \dots, x_n)$, as desired. Similarly considering $(H_\Sigma \pi_2 \circ \rho)(x, y)$ yields item 2.

($\Leftarrow$) Suppose we have a relation satisfying the latter condition, and define $\rho : R \to H_\Sigma R$ by $\rho(x, y) = f((x_1, y_1), \dots, (x_n, y_n))$. Then by item 1, $\epsilon_X \circ d \circ \pi_1 = \epsilon_X \circ H_\Sigma \pi_1 \circ \rho$, and similarly for item 2. □

Theorem 2 gives an explicit characterization for bisimulations up to presentation. To check that a relation is a bisimulation up to presentation, for each pair $(x, y)$ in the relation we need to rewrite $d(x)$ and $d(y)$ using $\epsilon$ laws so that they have the same symbol and all corresponding variables are related.

We can now show $x$ and $q$ from Example 9 are related by a bisimulation up to presentation.

**Example 10.** Recall that the zip functor has function symbols $\Sigma = \Sigma_2 = \{0\text{:zip}, 1\text{:zip}\}$ with no nontrivial $\epsilon$ laws. Then for the specification

$$\begin{array}{ccc} d(x) = 0\text{:zip}(y, z) & d(y) = 1\text{:zip}(x, v) & d(v) = 0\text{:zip}(w, u) \\ d(w) = 0\text{:zip}(w, z) & d(z) = 1\text{:zip}(z, w) & d(u) = 1\text{:zip}(u, v) \\ & d(q) = 0\text{:zip}(y, u) & \end{array}$$

we propose $R = \{(x, q), (z, u), (w, v)\} \cup \Delta_X$ as a bisimulation up to presentation. The diagonal part clearly satisfies the required properties. Then

- $0\text{:zip}(y, z) \sim_R 0\text{:zip}(y, u)$ since $yRy$ and $zRu$.
- $1\text{:zip}(z, w) \sim_R 1\text{:zip}(u, v)$ since $zRu$ and $wRv$.
- $0\text{:zip}(w, z) \sim_R 0\text{:zip}(w, u)$ since $wRw$ and $zRu$.

Since the zip functor has no nontrivial $\epsilon$ laws this is just an ordinary $H_\Sigma$-bisimulation. Matters are more complicated for non-polynomial functors.

**Example 11.** Recall that the functor $\mathcal{P}_3$ from Example 4 has a presentation with three function symbols, $\{\sigma_i\}_{i<3}$, each with arity $i$ and $\epsilon$ laws of the forms $\sigma_2(x, y) =_\epsilon \sigma_2(y, x)$ and $\sigma_2(x, x) =_\epsilon \sigma_1(x)$.

An example specification in this signature for $X = \{x, y, z\}$ might be

$$d(x) = \sigma_2(x, y) \qquad d(y) = \sigma_1(z) \qquad d(z) = \sigma_2(z, z)$$

All of these are behaviourally equivalent, so they should be related by a bisimulation up to presentation. We propose $R = \{(x, y), (y, z), (x, z), (z, z)\}$. For this we need to check four things:

- $\sigma_2(x, y) \sim_R \sigma_1(z)$: we use $\sigma_1(z) =_\epsilon \sigma_2(z, z)$ to rewrite the RHS and note $xRz$ and $yRz$.
- $\sigma_1(z) \sim_R \sigma_2(z, z)$: uses the same rewrite and $zRz$ twice.
- $\sigma_2(x, y) \sim_R \sigma_2(z, z)$: immediate from $xRz$ and $yRz$.
- $\sigma_2(z, z) \sim_R \sigma_2(z, z)$: immediate from $zRz$.

Therefore all three of these variables are related by a bisimulation up to presentation.

Finally we consider an example from a functor which does not preserve weak pullbacks.

**Example 12.** Recall the valuation functor $\mathbb{K}_\omega^{Id}$ from Example 6 has signature $\Sigma_n = \mathbb{K}^n$ and transformation $\epsilon_X : (l_1, \dots, l_n)(x_1, \dots, x_n) \mapsto \lambda x. \sum_{i:x_i=x} l_i$ for $n \geq 1$ and $\epsilon_X : () \mapsto \lambda x.0$ for $n = 0$.

An example specification on the variables $X = \{x, y, z, w\}$ in this signature:

$$d(x) = (-1, 1)(y, z) \qquad d(y) = () \qquad d(z) = () \qquad d(w) = ()$$

Then $R = \{(x, w), (z, y)\}$ is a bisimulation up to presentation because

- $d(x) = (-1, 1)(y, z)\ c(R)\ (-1, 1)(y, y) =_\epsilon () = d(w)$
- $d(y) = () = d(z)$

### 3.2. Enhanced bisimulation up to presentation

In Example 10, we showed two variables in a specification were related by a bisimulation up to presentation, but in the course of this proof we added in the diagonal relation to make the bisimulation hypothesis go through. This is reminiscent of other combination bisimulation up to techniques, such as those studied by Rot et al. in [16]. In this section, we provide several enhancements to the bisimulation up to presentation technique which will be useful in the sequel.

First we note bisimulation up to presentation interacts well with union of bisimulations.

**Lemma 1.** *Suppose $(X, d)$ is a specification for a finitary functor presented by $(\Sigma, \epsilon)$. Let $S$ be any bisimulation on $X$, $T$ be a relation containing $S$, and $R$ be a relation on $X$ such that $xRy$ implies $d(x) \sim_T d(y)$. Then $R \cup S$ also has the property that $(x, y) \in R \cup S$ implies $d(x) \sim_T d(y)$.*

**Proof.** Since $S$ is a bisimulation on $X$, it is also a bisimulation up to presentation by Theorem 1. Then by Theorem 2, $xSy$ implies $d(x) \sim_S d(y)$. Since $S \subseteq T$, we have $c(S) \subseteq c(T)$ and therefore $\sim_S \subseteq \sim_T$. Hence $xSy$ implies $d(x) \sim_T d(y)$. Combining this with the hypothesis on $R$, we have the desired result. □

**Corollary 2.** *Recall that $\sim$ is the biggest $F$-bisimulation on $X$. If $R$ is a relation on a specification such that any of the following hold:*

- $xRy \to d(x) \sim_{R\cup\Delta_X} d(y)$
- $xRy \to d(x) \sim_{R\cup\sim} d(y)$

*then $R \subseteq \sim$.*

**Proof.** Both $\Delta_X$ and $\sim$ are bisimulations so by Lemma 1, taking $T = R \cup B$ where $B \in \{\Delta_X, \sim\}$, we get a relation $T$ such that $xTy \to d(x) \sim_T d(y)$. Then $T$ is a bisimulation up to presentation by Theorem 2, so $R \subseteq T \subseteq \sim$. □

Bisimulation up to presentation also behaves well with respect to symmetric closures:

**Lemma 2.** *Suppose $(X, d)$ is a specification for a finitary functor presented by $(\Sigma, \epsilon)$. Let $T$ be any symmetric relation and $R$ be any relation on $X$ such that $xRy$ implies $d(x) \sim_T d(y)$. Then $s(R)$, the symmetric closure of $R$, also has the property $x\ s(R)\ y$ implies $d(x) \sim_T d(y)$.*

**Proof.** It is easy to check that $T$ symmetric implies $c(T)$ symmetric, which in turn implies $\sim_T$ symmetric. Then if $x\ s(R)\ y$, either $xRy$ or $yRx$ by definition of $s(R)$. The hypothesis on $R$ yields $d(x) \sim_T d(y)$ or $d(y) \sim_T d(x)$, respectively. Then $\sim_T$ symmetric allows us to conclude that in either case $d(x) \sim_T d(y)$, as desired. □

**Corollary 3.** *If $R$ is a relation such that $xRy \to d(x) \sim_{s(R)} d(y)$, then $R \subseteq \sim$.*

Bisimulation up to presentation for functors *preserving weak pullbacks* also plays well with equivalence closures. Preservation of weak pullbacks is a critical assumption here. We recall the following definition and theorem from [17, p.14], slightly recast to use our notation:

**Definition 3.** A presentation is *dominated* if for every $\epsilon$ law $f(\vec{x}) =_\epsilon g(\vec{y})$ where $f \in \Sigma_n$, $g \in \Sigma_m$, $\vec{x} : n \to X$, $\vec{y} : m \to X$, there is a symbol $h \in \Sigma_k$ and functions $u : k \to n$ and $v : k \to m$ such that $h(u) =_\epsilon f(id_n)$, $h(v) =_\epsilon g(id_m)$, and $\vec{x} \circ u = \vec{y} \circ v$.

To give some intuition for dominated presentations, consider the presentation for $\mathcal{P}_\omega$ from Example 3. We could say the symbol $\sigma_4$ dominates the symbol $\sigma_2$ since there is a $u : 4 \to 2$, namely $u(i) = \lfloor \frac{i}{2} \rfloor$, such that $\sigma_4(u) = \sigma_4(0, 0, 1, 1) =_\epsilon \sigma_2(0, 1) = \sigma_2(id_2)$. Then any time we have a term using $\sigma_2$, we could replace $\sigma_2$ with $\sigma_4$, following the substitution scheme hinted at by $u$, and remain in the same component of the kernel of $\epsilon$. So, for example, this domination would imply $\sigma_2(x, y) =_\epsilon \sigma_4(x, x, y, y)$.

Note $\sigma_3$ is also dominated by $\sigma_4$, for example by $\sigma_3(0, 1, 2) = \sigma_4(0, 2, 1, 1)$. This would allow us to rewrite $\sigma_3(x, y, x) =_\epsilon \sigma_4(x, x, y, y)$. Combining with the previous paragraph would allow us to derive $\sigma_3(x, y, x) =_\epsilon \sigma_2(x, y)$ via rewrites to $\sigma_4$. We can then say that $\sigma_3(x, y, x) =_\epsilon \sigma_2(x, y)$ is a consequence of the joint domination of $\sigma_2$ and $\sigma_3$ by $\sigma_4$.[6]

The verbiage for all this notation then is that a presentation is dominated means for every $\epsilon$ law $f(\vec{x}) =_\epsilon g(\vec{y})$ there is a dominating symbol $h$ with two variable substitutions $u$ and $v$ such that the $\epsilon$ law is a consequence of the joint domination of $f$ and $g$ by $h$ via the substitutions $u$ and $v$. We will be relying on facts about dominated presentations only in the proof of Lemma 3.

**Theorem 3** *(Adámek, Gumm, Trnková [17]). A finitary functor weakly preserves pullbacks if and only if it has a dominated presentation.*

[6] Obviously, the joint domination is not unique in the case of $\mathcal{P}_\omega$. $\sigma_2$ and $\sigma_3$ are jointly dominated by $\sigma_i$ for all $i \geq 3$ and even for each dominating symbol there may be many different substitutions which yield the desired equation as a consequence of the joint domination.

As a result of this theorem, for the next lemmas we can assume our presentation is dominated without loss of generality.

**Lemma 3.** *Suppose $T$ is an equivalence relation and $(\Sigma, \epsilon)$ is a dominated presentation. Then $\sim_T$ is an equivalence relation.*

**Proof.** Reflexivity and symmetry are straightforward. $T$ is reflexive and symmetric, therefore $c(T)$ is reflexive and symmetric, and so $\sim_T$ is reflexive and symmetric.

Transitivity requires the dominated presentation. Suppose $\alpha \sim_T \beta \sim_T \gamma$. Then we can write

$$\alpha =_\epsilon \alpha' \; c(T) \; \beta' =_\epsilon \beta =_\epsilon \beta'' \; c(T) \; \gamma' =_\epsilon \gamma$$

Let $\beta' = f(\vec{x})$ and $\beta'' = g(\vec{y})$. Then the above relations become:

$$\alpha =_\epsilon f(\vec{x'}) \; c(T) \; f(\vec{x}) =_\epsilon g(\vec{y}) \; c(T) \; g(\vec{y'}) =_\epsilon \gamma$$

Since we have a dominated presentation, we get $h$, $u$, and $v$ such that $h(u) =_\epsilon f(id_n)$, $h(v) =_\epsilon g(id_m)$ and $\vec{x} \circ u = \vec{y} \circ v$. The first two statements imply $f(\vec{x'}) =_\epsilon h(\vec{x'} \circ u)$ and $g(\vec{y'}) =_\epsilon h(\vec{y'} \circ v)$.

We also know $\vec{x'}(i) \; T \; \vec{x}(i)$ for all $i \in [1, n]$ and $\vec{y}(i) \; T \; \vec{y}'(i)$ for $i \in [1, m]$. Therefore, $(\vec{x'} \circ u)(i) \; T \; (\vec{x} \circ u)(i) = (\vec{y} \circ v)(i) \; T \; (\vec{y'} \circ v)(i)$ for $i \in [1, k]$, where the middle equality is by the last condition guaranteed by the dominated presentation. Then since $T$ is transitive we know $(\vec{x'} \circ u)(i) \; T \; (\vec{y'} \circ v)(i)$ for $i \in [1, k]$.

Therefore we have produced terms such that

$$\alpha =_\epsilon h(\vec{x'} \circ u) \; c(T) \; h(\vec{y'} \circ v) =_\epsilon \gamma$$

and so $\sim_T$ is transitive and hence is an equivalence relation. □

**Lemma 4.** *Suppose $(X, d)$ is a specification for a finitary functor preserving weak pullbacks with the dominated presentation $(\Sigma, \epsilon)$. Let $T$ be any equivalence relation on $X$ and $R$ be any relation on $X$ such that $xRy$ implies $d(x) \sim_T d(y)$. Then $e(R)$, the equivalence closure of $R$, has the property $x \, e(R) \, y$ implies $d(x) \sim_T d(y)$.*

**Proof.** By Lemmas 1 and 2, we know immediately that $x \; sr(R) \; y$ implies $d(x) \sim_T d(y)$, where $sr(R)$ is the symmetric reflexive closure of $R$. Hence we only have to consider $(x, y) \in e(R) \setminus sr(R)$. Therefore, suppose we have $x \; sr(R) \; z \; sr(R) \; y$. Then by the noted property of $sr(R)$ we get $d(x) \sim_T d(z) \sim_T d(y)$. By the previous Lemma, since $T$ is an equivalence relation and we have a dominated presentation, $\sim_T$ is an equivalence relation and hence $d(x) \sim_T d(y)$. □

**Corollary 4.** *Suppose $F$ preserves weak pullbacks and its presentation $(\Sigma, \epsilon)$ is dominated. If $R$ is a relation such that $xRy$ implies $d(x) \sim_{e(R)} d(y)$, then $R \subseteq \sim$.*

The following corollary follows directly from the results above, but is of critical importance to our proof of soundness.

**Corollary 5.** *Suppose $F$ preserves weak pullbacks and its presentation $(\Sigma, \epsilon)$ is dominated. If $R$ is a relation such that $xRy$ implies $d(x) \sim_{e(R \cup \sim)} d(y)$, then $R \subseteq \sim$.*

## 4. A proof system for bisimulation up to presentation

In this section, we outline a formal proof system to capture the notion of bisimulation up to presentation. For this whole section, we assume $F$ preserves weak pullbacks. We then prove this system to be sound and complete. Our system has judgements of the form $R \vdash \sigma = \tau$ where $R \subseteq X \times X$ and $(\sigma, \tau) \in X \times X + H_\Sigma X \times H_\Sigma X$. The inference rules are as follows:

$$\frac{}{R \vdash \sigma = \sigma}\; r \qquad \frac{R \vdash \tau = \sigma}{R \vdash \sigma = \tau}\; s \qquad \frac{R \vdash \sigma = \tau \quad R \vdash \tau = \rho}{R \vdash \sigma = \rho}\; t$$

$$\frac{}{\{(x, y)\} \cup R \vdash x = y}\; a \qquad \frac{R \vdash x_1 = y_1 \quad \dots \quad R \vdash x_{ar(f)} = y_{ar(f)}}{R \vdash f(x_1, \dots, x_{ar(f)}) = f(y_1, \dots, y_{ar(f)})}\; c$$

$$\frac{\sigma =_\epsilon \tau}{R \vdash \sigma = \tau}\; \epsilon \qquad \frac{R \vdash \varphi \quad \forall (x, y) \in R. R \vdash d(x) = d(y)}{\vdash \varphi}\; b$$

As usual, we say $R \vdash \varphi$ when there is a proof tree using the above rules with the judgement $R \vdash \varphi$ as the root. The notation $\vdash \varphi$ is shorthand for $\varnothing \vdash \varphi$. Recall that $\vDash x = y$ means that $x$ and $y$ are behaviourally equivalent (have the same image in the final coalgebra).

We should point out that $R$ on the left side of the turnstile does not have the usual force of a full assumption. Rather, this $R$ should be thought of as tracking an unverified bisimulation hypothesis. In most rules this unverified bisimulation remains unchanged across the inference. The notable exception is the $b$ rule, which essentially discharges a verified bisimulation. This $b$ rule is really just the coinductive proof principle. We also note its similarity to the Recursion Inference Rule from $\mathrm{FLR}_0$ [18], which was in mind as the system was constructed.

Before we prove soundness and completeness, we give two example proofs using the system. This first example is based on Example 4.2 in Moss et al. [3].

**Example 13.** Consider the specification on $X = \{x, y, r, s\}$ for $\mathcal{P}_3$ defined by

$$d(x) = \sigma_2(x, y) \qquad d(y) = \sigma_0 \qquad d(r) = \sigma_2(r, s) \qquad d(s) = \sigma_0$$

Let $R = \{(x, r), (y, s)\}$. The proof tree below witnesses $\vdash x = r$.

$$\dfrac{\dfrac{}{R \vdash x = r}\,a \qquad \dfrac{\dfrac{}{R \vdash x = r}\,a \quad \dfrac{}{R \vdash y = s}\,a}{R \vdash \sigma_2(x, y) = \sigma_2(r, s)}\,c \qquad \dfrac{}{R \vdash \sigma_0 = \sigma_0}\,c}{\vdash x = r}\,b$$

The next example is adapted from Example 11 in this paper and showcases how some rules allow for shorter proofs.

**Example 14.** Consider the specification on $X = \{x, y\}$ for $\mathcal{P}_3$ defined by

$$d(x) = \sigma_2(x, y) \qquad d(y) = \sigma_1(x)$$

Let $R = \{(x, y)\}$. The proof tree below witnesses $\vdash x = y$.

$$\dfrac{\dfrac{}{R \vdash x = y}\,a \qquad \dfrac{\dfrac{\dfrac{}{R \vdash x = x}\,r \quad \dfrac{\dfrac{}{R \vdash x = y}\,a}{R \vdash y = x}\,s}{R \vdash \sigma_2(x, y) = \sigma_2(x, x)}\,c \qquad \dfrac{}{R \vdash \sigma_2(x, x) = \sigma_1(x)}\,\epsilon}{R \vdash \sigma_2(x, y) = \sigma_1(x)}\,t}{\vdash x = y}\,b$$

Note that $R$ contains a single pair though an unenhanced bisimulation proof would require three: the $r$ rule allows us to omit $(x, x)$, and the $s$ rule elides the mirror-image proof that $d(y) = d(x)$ thereby allowing us to omit $(y, x)$.

### *4.1. Soundness*

To help with our soundness proof, we define a new relational closure which we call the *presentational closure* of a relation $R \subseteq X \times X$ on the carrier of a specification $(X, d)$. Recall $\sim_R$ is defined to be $=_\epsilon \bullet c(R) \bullet =_\epsilon$. That is, $\alpha \sim_R \beta$ if we can rewrite $\alpha$ and $\beta$ using $\epsilon$ laws so they have the same function symbol and all corresponding variables are related by $R$. The presentational closure of $R$, denoted $pr(R)$, is defined to be $pr(R) \triangleq e(R \cup \sim) \cup \sim_{e(R \cup \sim)}$, a relation on $X \cup H_\Sigma X$.

We note that $e(R \cup \sim)$ is an equivalence relation on $X$, and $\sim_{e(R\cup\sim)}$ is an equivalence relation on $H_\Sigma X$ as a consequence. Since $X$ and $H_\Sigma X$ are disjoint, $pr(R)$ is also an equivalence relation.

**Proposition 1.** *If $R \vdash \varphi$, then $\varphi \in pr(R)$.*

**Proof.** By induction on the proof tree. The base cases are $r$, $\epsilon$, and $a$. We know $pr(R)$ is an equivalence relation, hence $r$. The relation $e(R \cup \sim)$ is reflexive, hence $\sim_{e(R\cup\sim)}$ contains the relation $=_\epsilon$, hence $\epsilon$. All pairs in $R$ are included in $e(R \cup \sim)$, hence $a$.

The induction steps are $s$, $t$, $c$, and $b$. The first two, $s$ and $t$, follow easily from the fact that $pr(R)$ is an equivalence relation.

The induction hypothesis for the $c$ rule states for all $1 \le i \le ar(f)$, we know $(x_i, y_i) \in pr(R)$ and hence $(x_i, y_i) \in e(R \cup \sim)$. Then by definition of the flat contextual closure $f(x_1, \ldots, x_{ar(f)})\ c(e(R \cup \sim))\ f(y_1, \ldots, y_{ar(f)})$. Since $=_\epsilon$ is a reflexive relation, $pr(R)$ contains $c(e(R \cup \sim))$ and hence these two terms are related by $pr(R)$. Therefore the induction holds across the $c$ rule.

Finally we consider the $b$ rule. The induction hypothesis gives $\varphi \in pr(R)$ and $(d(x), d(y)) \in pr(R)$ for each $(x, y) \in R$. Since $d(x)$ and $d(y) \in H_\Sigma X$ we know $d(x) \sim_{e(R\cup\sim)} d(y)$. Then by Corollary 5, we know $R \subseteq \sim$. Therefore, $pr(R) = e(R \cup \sim) \cup \sim_{e(R\cup\sim)} = e(\sim) \cup \sim_{e(\sim)} = pr(\varnothing)$. Then $\varphi \in pr(\varnothing)$ since $\varphi \in pr(R)$. $\square$

**Corollary 6** *(Soundness). If $x, y \in X$ and $\vdash x = y$, then $\vDash x = y$.*

**Proof.** If $\vdash x = y$, then $(x, y) \in pr(\varnothing)$ by the previous proposition. Since $pr(\varnothing) \triangleq e(\sim \cup \varnothing) \cup \sim_{e(\sim\cup\varnothing)}$ and $\sim_{e(\sim\cup\varnothing)}$ is a relation strictly on $H_\Sigma X$, we know $(x, y) \in e(\sim \cup \varnothing)$. However, clearly $e(\sim \cup \varnothing) = e(\sim) = \sim$, so $x \sim y$. A standard fact about functors preserving weak pullbacks is that two states in a coalgebra are bisimilar if and only if they are behaviourally equivalent [9], so $x \sim y$ implies $\vDash x = y$. $\square$

### 4.2. Completeness

The proof of completeness is much more straightforward but also critically relies on the functor preserving weak pullbacks.

**Lemma 5.** *If $\alpha \sim_R \beta$, then $R \vdash \alpha = \beta$.*

**Proof.** $\alpha \sim_R \beta$ means there are $f \in \Sigma_n$ and $(x_1, y_1), \ldots, (x_n, y_n) \in R$ such that

$$\alpha =_\epsilon f(x_1, \ldots, x_n)\, c(R)\, f(y_1, \ldots, y_n) =_\epsilon \beta.$$

Then

$$\dfrac{\dfrac{}{R \vdash \alpha = f(x_1,\ldots,x_n)}\,\epsilon \quad \dfrac{\dfrac{}{R\vdash x_1 = y_1}\,a \quad \cdots \quad \dfrac{}{R \vdash x_n = y_n}\,a}{R \vdash f(x_1,\ldots,x_n) = f(y_1,\ldots,y_n)}\,c}{\dfrac{R \vdash \alpha = f(y_1,\ldots,y_n)}{} \,t \quad \dfrac{}{R \vdash f(y_1,\ldots,y_n) = \beta}\,\epsilon}$$

$$\dfrac{R \vdash \alpha = f(y_1,\ldots,y_n) \qquad R \vdash f(y_1,\ldots,y_n) = \beta}{R \vdash \alpha = \beta}\,t$$

is a witness for $R \vdash \alpha = \beta$. $\square$

**Corollary 7** *(Completeness). If $\vDash x = y$, then $\vdash x = y$.*

**Proof.** Recall that $\vDash x = y$ iff $x \sim y$. Since $x$ and $y$ are bisimilar, they are related by a bisimulation up to presentation, which we call $R$. By Theorem 2, $uRv \to d(u) \sim_R d(v)$. Syllogizing with the previous lemma yields $uRv \to (R \vdash d(u) = d(v))$. Trivially, $R \vdash x = y$ by the $a$ rule. Therefore, $\vdash x = y$ by the $b$ rule. $\square$

## 5. Compositionality of presentations

Bisimulations up to presentation give a uniform way to reason about behavioural equivalence of variables in specifications for finitary functors. In this section we show this proof system is sound, complete, and **compositional**. That is, the rules for reasoning about coalgebras of a functor are built inductively in a manner corresponding to the definition of the functor. The prime example of this situation is that of the polynomial functors and the Kripke polynomial functors.

A functor is called *polynomial* if it is generated by the following BNF grammar:

$$P ::= A \mid Id \mid P + P \mid P \times P \mid P^B$$

where $A$ is the constant functor having value $A \in Set$ and $B$ is a finite set. The *Kripke polynomial* class of functors adds the finite powerset functor:

$$K ::= A \mid Id \mid P_\omega(K) \mid K + K \mid K \times K \mid K^B$$

Bonsangue et al. build a sound, complete and compositional expression calculus to represent coalgebras of Kripke polynomial functors in [5]. We show presentations are similarly compositional, in that both the signature and the $\epsilon$ transformation can be built inductively to parallel the construction of the functor. For the following three constructions, suppose $F$ and $G$ are finitary functors with presentations $(\Sigma, \epsilon)$ and $(\Sigma', \epsilon')$.

### 5.1. Products

Let $J = F \times G$. Then we claim $J$ has a presentation $(\Sigma'', \epsilon'')$ where $\Sigma''$ has all pairs of symbols, $\Sigma''_n = \{(f, g) : f \in \Sigma, g \in \Sigma', ar(f) + ar(g) = n\}$, and $\epsilon'' : H_{\Sigma''} \to J$ has components

$$\epsilon''_X : (f, g)(x_1, \ldots, x_n) \mapsto (\epsilon_X(f(x_1, \ldots, x_{ar(f)})), \epsilon'_X(g(x_{ar(f)+1}, \ldots, x_n))).$$

Then $\epsilon''$ is an epic natural transformation as a consequence of $\epsilon$ and $\epsilon'$ being epic natural transformations.

We single out this particular presentation because it allows us to state the $\epsilon''$ laws in terms of $\epsilon$ and $\epsilon'$ laws. By definition $\epsilon''((f, g)(x_1, \ldots, x_n)) = \epsilon''((f', g')(y_1, \ldots y_m))$ means

$$\epsilon(f(x_1, \ldots, x_{ar(f)})) = \epsilon(f'(y_1, \ldots, y_{ar(f')})) \text{ and}$$

$$\epsilon'(g(x_{ar(f)+1}, \ldots, x_n)) = \epsilon'(g'(y_{ar(f')+1}, \ldots, y_m)).$$

Therefore, $\epsilon''$ laws in this presentation are pairs of $\epsilon$ and $\epsilon'$ laws.

We note here that we could represent finite powers with a similar construction. If $B$ is a finite set, a signature for $F^B$ has symbols $|B|$ tuples of symbols from $\Sigma$ with arity the sum of the arities through the tuple. Then the $\epsilon''$ laws are $|B|$ tuples of $\epsilon$ laws.

### 5.2. Coproducts

Let $J = F + G$. We write inl and inr for the standard coproduct injections. Then $J$ has a presentation $(\Sigma'', \epsilon'')$ where $\Sigma''_n = \Sigma_n + \Sigma'_n$ and $\epsilon''$ has components $\epsilon''_X$ such that

$$\begin{cases} \epsilon''_X(\mathsf{inl} f(\vec{x})) = \mathsf{inl}\epsilon_X(f(\vec{x})) \\ \epsilon''_X(\mathsf{inr} g(\vec{x})) = \mathsf{inr}\epsilon'_X(g(\vec{x})) \end{cases}$$

Since $\epsilon$ and $\epsilon'$ are epic natural transformations, $\epsilon''$ is also an epic natural transformation.

Again, we can state the $\epsilon''$ laws in terms of the $\epsilon$ and $\epsilon'$ laws. By definition, $\epsilon''\alpha = \epsilon''\beta$ means $\alpha$ and $\beta$ are both labelled inl or are both labelled inr. In the former case, we have $\mathsf{inl}\epsilon(f(\vec{x})) = \mathsf{inl}\epsilon(g(\vec{y}))$, which is an inl-labelled instance of an $\epsilon$-law. Similarly, the latter case gives an inr-labelled instance of an $\epsilon'$-law.

### 5.3. Compositions

Let $J = G \circ F$. Then $J$ has a presentation with symbols from the set $\Sigma'' = \{(\sigma', (\sigma_1, \ldots, \sigma_{ar(\sigma')})) : \sigma' \in \Sigma' \text{ and } \sigma_i \in \Sigma\}$. For each symbol $\sigma'' \in \Sigma''$ define $w_{\sigma''}(i) = \sum_{j=1}^{i} ar(\sigma_i)$ for $0 \le i \le ar(\sigma')$ and define $\sigma''$ to have arity $w_{\sigma''} = w_{\sigma''}(ar(\sigma'))$. Given an $ar(\sigma'')$ tuple from $X$, we let $\vec{x}_i = (x_{w_{\sigma''}(i-1)+1}, \ldots, x_{w_{\sigma''}(i)})$ for $1 \le i \le ar(\sigma')$, the slice of the variables corresponding to $\sigma_i$. The natural transformation $\epsilon''$ has components given by

$$\epsilon'' : \sigma''(x_1, \ldots, x_{ar(\sigma'')}) \mapsto \epsilon'(\sigma'(\epsilon(\sigma_1(\vec{x}_1)), \ldots, \epsilon(\sigma_{ar(\sigma')}(\vec{x}_{ar(\sigma'')})))).$$

This is an epic natural transformation and the $\epsilon''$ laws can be stated again in terms of the $\epsilon$ and $\epsilon'$ laws.

### 5.4. Kripke polynomial functors and other polynomial-like classes of functors

We have presentations for constant functors (Example 1), the identity functor (Example 2) and the finite power set functor (Example 3), which means the above constructions give a compositional presentation for each of the Kripke polynomial functors. Due to the previous section, we know bisimulation up to those presentations is a sound and complete proof system.

**Example 15.** In Example 6 we gave a presentation for the valuation functor $\mathbb{K}^{Id}_{\omega}$. This functor is related to an important functor $\mathcal{W} = \mathbb{K} \times \mathbb{K}^{Id}_{\omega}$ whose coalgebras are weighted automata. The observations above allow us to construct a presentation for $\mathcal{W}$ from the presentations for the constant functor $\mathbb{K}$ and the valuation functor: the signature for $\mathcal{W}$ is given by $\Sigma_n = \mathbb{K}^{n+1}$ and has equations of the form $(l_0, l_1, \ldots, l_n)(x_1, \ldots x_n) =_\epsilon (k_0, k_1, \ldots, k_m)(y_1, \ldots, y_m)$ if and only if $l_0 = k_0$ and $(l_1, \ldots, l_n)(x_1, \ldots x_n)$ and $(k_1, \ldots, k_m)(y_1, \ldots, y_m)$ are identified in the presentation of $\mathbb{K}^{Id}_{\omega}$. To be a bit more explicit, this means $l_0 = k_0$ and $\sum_{i:x_i=x} l_i = \sum_{j:y_j=x} k_j$ for all $x \in X$.

**Example 16.** The functor $F = A \times \mathrm{Id}^B$ where $B$ is finite also has applications in automata theory since its coalgebras are Moore machines. Following the constructions above starting with the constant and identity functors, $F$ has a presentation with function symbols of arity $|B|$ given by $(a, \overbrace{*, \ldots, *}^{|B|})$ for $a \in A$. This presentation has equations $(a, *, \ldots, *)(x_1, \ldots, x_{|B|}) =_\epsilon (a', *, \ldots, *)(y_1, \ldots, y_{|B|})$ iff $a = a'$ and $x_i = y_i$ for all $1 \le i \le |B|$.

The presentation above follows the recipe in this section fairly strictly, but obviously the simplified signature replacing "$(a, *, \ldots, *)$" with "$a$" (still with arity $|B|$) would make for much more readable specifications.

## 6. Removing the weak pullback requirement

Finitary functors enjoy another special property: their final coalgebra structure can be explicitly characterized from the presentation. Adámek and Milius characterize the final $F$-coalgebra in [19] as the set of finite or infinite $\Sigma$ trees up to finite or "infinite" application of $\epsilon$ laws. We briefly recount their definitions and results in Section 6.1, then use those facts to prove soundness and conclude with a few examples.

### 6.1. An explicit description of the final $F$-coalgebra

Let $T_\Sigma$ be the set of all finite or infinite $\Sigma$ trees. These trees must be closed: all leaves in a tree from $T_\Sigma$ are nullary symbols from $\Sigma$ and if $\Sigma$ has no nullary symbols, every tree in $T_\Sigma$ is infinite. This set carries the final $H_\Sigma$-coalgebra with a well-known final map. Given a specification $(X, d)$, the final map to $T_\Sigma$ takes $x \in X$ to its "tree unfolding". That is, we start with the singleton tree with node labelled $x$ and at each step replace each leaf node labelled $y \in X$ by the tree $d(y)$ of height one. This procedure guarantees after $k$ steps the leaves at level $k$ are variables and all leaves and internal nodes at levels $< k$ are from $\Sigma$ and so do not change in subsequent replacement steps. Therefore, this sequence of trees has a limit which we call the *tree unfolding of $x$ (along $d$)*, denoted $t_d(x)$.

We let $\phi : T_\Sigma \to H_\Sigma T_\Sigma$ give the $H_\Sigma$-coalgebra structure for $T_\Sigma$. $\phi$ is basically a tree destructor; given a tree $\tau \in T_\Sigma$, $\phi(\tau)$ returns the element $f \in \Sigma$ at $\tau$'s root along with the $ar(f)$ subtrees below the root.

The equation $\phi \circ t_d = H_\Sigma t_d \circ d$ expresses the fact that $t_d$ is the final $H_\Sigma$-coalgebra map from the specification $(X, d)$. (We will use this in the proof of Lemma 6 below.)

Using the transformation $\epsilon$, we can make $T_\Sigma$ into an $F$-coalgebra by postcomposition with $\epsilon_{T_\Sigma}$. In fact, this makes $T_\Sigma$ a *weakly* final $F$-coalgebra, in the sense that there always exists a (not necessarily unique) $F$-coalgebra morphism into $T_\Sigma$. To see this, given an $F$-coalgebra $f : X \to FX$, use any section of $\epsilon_X$ to form the related specification $s \circ f = d : X \to H_\Sigma X$. This specification immediately yields a $H_\Sigma$-coalgebra morphism into $T_\Sigma$, namely tree unfolding, which is the left square below. The naturality of $\epsilon$ means the right square commutes, and these facts taken together show that tree unfolding is actually also an $F$-coalgebra morphism.

$$\begin{tikzcd} X \arrow[r, "d"] \arrow[d, "t_d"'] & H_\Sigma X \arrow[r, "\epsilon_X"] \arrow[d, "H_\Sigma t_d"] & FX \arrow[d, "F t_d"] \\ T_\Sigma \arrow[r, "\phi"'] & H_\Sigma T_\Sigma \arrow[r, "\epsilon_{T_\Sigma}"'] & F T_\Sigma \end{tikzcd}$$

Different sections of $\epsilon_X$ give rise to different specifications $d$ which in turn yield potentially different tree unfoldings. However, these tree unfoldings still have some common structure—at each leaf replacement step the possible replacements for $x$ are all related by $=_\epsilon$. Indeed, by quotienting $T_\Sigma$ by a relation allowing $=_\epsilon$-related subtrees to be substituted, this coalgebra becomes final in $\mathbf{Coalg}_F$.

We describe $\approx_\epsilon$, a relation on $T_\Sigma$ extending the relation $=_\epsilon$ on $H_\Sigma T_\Sigma$, to make this more precise. For finite trees, $\approx_\epsilon$ is the least congruence relation generated by $=_\epsilon$. For infinite trees, $\sigma \approx_\epsilon \tau$ means each level cutting of $\sigma$ and $\tau$ are $\approx_\epsilon$ related. The level $(k)$ cutting of an infinite tree $\sigma \in T_\Sigma$ is defined to be the biggest height $k$ subtree of $\sigma$ with the level $k$ leaves replaced by a fresh symbol $\perp$.[7]

**Theorem 4** *(Adámek and Milius [19]). $T_\Sigma/\approx_\epsilon$ is the final $F$-coalgebra.*

In [19], it is also noted $\approx_\epsilon$ (written $\sim^*$ in that work) is a congruence relation, so there is a morphism $\chi$ making the following diagram commute:

$$\begin{tikzcd} T_\Sigma \arrow[r, "\phi"] \arrow[d, "q_{\approx_\epsilon}"'] & H_\Sigma T_\Sigma \arrow[r, "\epsilon_X"] \arrow[d, "H_\Sigma q_{\approx_\epsilon}"] & F T_\Sigma \arrow[d, "F q_{\approx_\epsilon}"] \\ T_\Sigma/\approx_\epsilon \arrow[rr, bend right, "\chi"'] & H_\Sigma(T_\Sigma/\approx_\epsilon) \arrow[r, "\epsilon_{T_\Sigma/\approx_\epsilon}"] & F(T_\Sigma/\approx_\epsilon) \end{tikzcd}$$

where $q_R$ is the quotient map from a set to its equivalence classes under the equivalence relation $R$.

Given a specification $(X, d)$, recall the relation $\vDash$ consists of pairs of variables which have the same image in the final coalgebra. By uniqueness of the final map, we know $m \circ q_\vDash = q_{\approx_\epsilon} \circ t_d$ for some monic $m$.

### 6.2. A sound calculus for behavioural equivalence

The explicit description of the final coalgebra map as a tree unfolding followed by a quotient process based on $=_\epsilon$ replacements gives us a direct route to determining whether two variables in a specification are behaviourally equivalent. In fact, the logic presented in Section 4 remains sound even without the assumption that the functor preserves weak pullbacks.

[7] We characterize this paragraph as a "description" rather than a "definition" since these level $k$ cuttings are not elements of $T_\Sigma$, but rather in $T_{\Sigma'}$, a related signature with this fresh nullary symbol $\perp$, so it is not quite precise. We will not rely on the precise definition of $\approx_\epsilon$ however, only the fact that it is a congruence.

To prove this, we point out a particular property of $\approx_\epsilon$. If we have two trees $\sigma, \tau \in T_\Sigma$ such that $\sigma \approx_\epsilon \tau$, then they are joined by $q_{\approx_\epsilon}$ in the lower left path in the diagram above. That means they have the same image in $F(T_\Sigma/\approx_\epsilon)$ and therefore $H_\Sigma q_{\approx_\epsilon}(\phi(\sigma)) =_\epsilon H_\Sigma q_{\approx_\epsilon}(\phi(\tau))$ by following the middle path above.

A similar property holds of $\vDash$, which we demonstrate next.

**Lemma 6.** *Let $(X, d)$ be a specification. If $R$ is any equivalence relation on $X$ which contains $\vDash$, then $H_\Sigma q_R(d(x)) =_\epsilon H_\Sigma q_R(d(y))$ for all $\vDash x = y$.*

**Proof.** Suppose $(x, y) \in \vDash$, so $t_d(x) \approx_\epsilon t_d(y)$. By the property of $\approx_\epsilon$ noted above, we know $H_\Sigma q_{\approx_\epsilon}(\phi(t_d(x))) =_\epsilon H_\Sigma q_{\approx_\epsilon}(\phi(t_d(y)))$. Previously noted equalities imply $H_\Sigma q_{\approx_\epsilon} \circ \phi \circ t_d = H_\Sigma q_{\approx_\epsilon} \circ H_\Sigma t_d \circ d = H_\Sigma(m \circ q_\vDash) \circ d$, so the above becomes $H_\Sigma q_\vDash(d(x)) =_\epsilon H_\Sigma q_\vDash(d(y))$ after cancelling the monic.

Since $\vDash \subseteq R$, we can find a $q$ to make $q_R = q \circ q_\vDash$. Then naturality of $\epsilon$ allows us to conclude $H_\Sigma q_R(d(x)) =_\epsilon H_\Sigma q_R(d(y))$ after applying $H_\Sigma q$ to both sides of $H_\Sigma q_\vDash(d(x)) =_\epsilon H_\Sigma q_\vDash(d(y))$. □

When a similar property holds of other relations, we can actually show that it is a subset of the $\vDash$ relation.

**Lemma 7.** *Let $(X, d)$ be a specification. If $R$ is a relation on $X$ such that $(x, y) \in R$ implies $H_\Sigma q_{e(R)}(d(x)) =_\epsilon H_\Sigma q_{e(R)}(d(y))$, then $R \subseteq \vDash$.*

**Proof.** The above condition on $R$ implies the two paths in the diagram below commute:

$$\begin{tikzcd}
R \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\pi_2"'] & X \arrow[r, "d"] & H_\Sigma X \arrow[d, "H_\Sigma q_{e(R)}"] & \\
& & H_\Sigma\ {}^{X}/_{e(R)} \arrow[r, "\epsilon_{X/e(R)}"] & F\, {}^{X}/_{e(R)}
\end{tikzcd}$$

However, $q_{e(R)}$ is the coequalizer of these $\pi_i$, so there is a $\chi$ which makes the following diagram commute:

$$\begin{tikzcd}
R \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\pi_2"'] & X \arrow[r, "d"] \arrow[d, "q_{e(R)}"] & H_\Sigma X \arrow[r, "\epsilon_X"] \arrow[d, "H_\Sigma q_{e(R)}"] & FX \arrow[d, "F q_{e(R)}"] \\
& {}^{X}/_{e(R)} \arrow[rr, bend right, "\chi"'] & H_\Sigma\ {}^{X}/_{e(R)} \arrow[r, "\epsilon_{X/e(R)}"] & F\, {}^{X}/_{e(R)}
\end{tikzcd}$$

This $\chi$ gives ${}^{X}/_{e(R)}$ an $F$-coalgebra structure and $q_{e(R)}$ is an $F$-coalgebra morphism. The morphisms into the final $F$-coalgebra therefore fill in a diagram with $q_{e(R)}$. Since $(x, y) \in R$ are joined by $q_{e(R)}$, $x$ and $y$ have the same image in the final coalgebra and hence $\vDash x = y$. □

Given a relation $R \subseteq X \times X$ on the variables in a specification $(X, d)$, we define $\overline{R} = e(R \cup \vDash)$. The following proposition is similar to Proposition 1, with $\overline{R}$ taking the role of $pr(R)$.

**Proposition 2.** *Given a specification $(X, d)$, let $x, y \in X$ be variables in this specification, and let $\sigma, \tau \in H_\Sigma X$ be flat terms in the signature. If $R \vdash x = y$, then $(x, y) \in \overline{R}$. If $R \vdash \sigma = \tau$, then $H_\Sigma q_{\overline{R}}(\sigma) =_\epsilon H_\Sigma q_{\overline{R}}(\tau)$.*

**Proof.** By induction on the proof tree which shows $R \vdash \alpha = \beta$.

Note both relations mentioned in the consequents above are equivalence relations, so the $r$, $s$ and $t$ rules are immediate. Since $R \subseteq \overline{R}$, the $a$ rule is also clear.

Application of the $\epsilon$ rule requires $\sigma =_\epsilon \tau$. By naturality of $\epsilon$, since $\sigma =_\epsilon \tau$ we know $H_\Sigma q_{\overline{R}}(\sigma) =_\epsilon H_\Sigma q_{\overline{R}}(\tau)$.

Next we consider the $c$ rule. The induction hypothesis gives $(x_i, y_i) \in \overline{R}$ for $1 \le i \le ar(f)$. Then $q_{\overline{R}}(x_i) = q_{\overline{R}}(y_i)$, so actually

$$\begin{aligned}
H_\Sigma q_{\overline{R}}(f(x_1, \dots, x_{ar(f)})) &= f(q_{\overline{R}}(x_1), \dots, q_{\overline{R}}(x_{ar(f)})) \\
&= f(q_{\overline{R}}(y_1), \dots, q_{\overline{R}}(y_{ar(f)})) \\
&= H_\Sigma q_{\overline{R}}(f(y_1, \dots, y_{ar(f)}))
\end{aligned}$$

which is stronger than $H_\Sigma q_{\overline{R}}(f(x_1, \dots, x_{ar(f)})) =_\epsilon H_\Sigma q_{\overline{R}}(f(y_1, \dots, y_{ar(f)}))$.

The $b$ rule is the final case. We know $H_\Sigma q_{\overline{R}}(d(x)) =_\epsilon H_\Sigma q_{\overline{R}}(d(y))$ for all $(x, y) \in R$ from the induction hypothesis. Since $\vDash \subseteq \overline{R}$, Lemma 6 allows us to strengthen this to $H_\Sigma q_{\overline{R}}(d(x)) =_\epsilon H_\Sigma q_{\overline{R}}(d(y))$ for all $(x, y) \in R \cup \vDash$. Then we can apply Lemma 7 to conclude that $R \subseteq \vDash$. Therefore $\overline{R} = \overline{\varnothing}$ which allows us to use the induction hypothesis from $R \vdash \varphi$ to show the corresponding statement holds of $\vdash \varphi$. □

Soundness of the proof system is a simple consequence of this proposition.

**Corollary 8.** *If $(X, d)$ is a specification, $x, y \in X$, and $\vdash x = y$, then $\vDash x = y$.*

**Proof.** Proposition 2 shows that if $\varnothing \vdash x = y$, then $(x, y) \in e(\varnothing \cup \vDash) = \vDash$. $\square$

### 6.3. Examples

We conclude this section with a few sample proofs that demonstrate how this proof system can be used with functors which do not preserve weak pullbacks.

The first example is extensively discussed in Section 2.3 of [20]. Let $\mathbb{K}$ be a semiring. The coalgebras of the finitary functor $\mathcal{W} = \mathbb{K} \times \mathbb{K}^{Id}_{\omega}$ are weighted automata: each state has an output weight and finitely many outgoing transitions each with a different target and each weighted by a value in $\mathbb{K}$.

We gave the presentation for this functor as Example 15. Quickly stated, function symbols of arity $n$ in this signature are lists of length $n+1$ from $\mathbb{K}$. The transformation maps $(k_0, k_1, \ldots, k_n)(x_1, \ldots, x_n)$ to $(k_0, \lambda x. \sum_{i:x_i=x} k_i)$.[8] As a result, a specification like $d(x) = (0, 1, 2)(y, z)$ will mean the state $x$ has the output weight 0, transitions to state $y$ with weight 1 and transitions to $z$ with weight 2, which is exactly the kind of transition structure expected in a weighted automaton.

Consider the following specification on $X = \{x, y, z, w\}$.

$$d(x) = (0, 1, -1)(y, z) \qquad d(z) = (1) \qquad d(y) = (1) \qquad d(w) = (0)$$

As noted in [20], there is no Aczel–Mendler bisimulation relating $x$ to $w$ in the related coalgebra for this specification. However, below is a proof in our system that these two states are behaviourally equivalent.[9] Take $R = \{(x, w), (y, z)\}$. Then

$$\dfrac{\dfrac{}{R \vdash x = w}\,a \qquad \dfrac{\dfrac{\dfrac{}{R \vdash y = z}\,a \quad \dfrac{}{R \vdash z = z}\,r}{R \vdash (0,1,-1)(y,z) = (0,1,-1)(z,z)}\,c \qquad \dfrac{}{R \vdash (0,1,-1)(z,z) = (0)}\,\epsilon}{R \vdash (0,1,-1)(y,z) = (0)}\,t}{\vdash x = w}\,b$$

(Technically $b$ requires that we also show $R \vdash d(y) = d(z)$. Since this claim is the trivial $R \vdash (1) = (1)$, we omitted it from the above tree to stay in the margins.)

The next example is for the Aczel–Mendler functor $(-)^3_2$, where $(X)^3_2 = \{(x, y, z) \in X^3 : |\{x, y, z\}| \le 2\}$. This functor does not preserve weak pullbacks, but is finitary with presentation given by $\Sigma = \Sigma_2 = \{\sigma, \tau, \rho\}$ and transformation

$$\epsilon_X : \sigma(x, y) \mapsto (x, x, y) \qquad \epsilon_X : \tau(x, y) \mapsto (y, x, x) \qquad \epsilon_X : \rho(x, y) \mapsto (x, y, x)$$

The only $\epsilon$ laws for this transformation are of the form $\sigma(x, x) = \tau(x, x) = \rho(x, x)$. Consider the following specification on $X = \{a, b, c\}$:

$$d(a) = \sigma(a, b) \qquad d(b) = \rho(a, a) \qquad d(c) = \tau(b, a)$$

The greatest bisimulation on the related coalgebra is not transitive. In particular, it does not relate $a$ and $c$.[10] However, $a$ and $c$ are behaviourally equivalent, as can be seen below. Take $R = \{(a, c), (a, b)\}$, then:

$$\dfrac{\dfrac{}{R \vdash a = c}\,a \qquad \dfrac{\dfrac{\dfrac{}{R \vdash a = a}\,r \quad \dfrac{\dfrac{}{R \vdash a = b}\,a}{R \vdash b = a}\,s}{R \vdash \sigma(a, b) = \sigma(a, a)}\,c \qquad \dfrac{}{R \vdash \sigma(a, a) = \rho(a, a)}\,\epsilon}{R \vdash \sigma(a, b) = \rho(a, a)}\,t \qquad \ldots}{\vdash a = c}$$

$$\dfrac{\ldots \qquad \dfrac{\dfrac{\dfrac{}{R \vdash a = a}\,r \quad \dfrac{\dfrac{}{R \vdash a = b}\,a}{R \vdash b = a}\,s}{R \vdash \sigma(a, b) = \sigma(a, a)}\,c \qquad \dfrac{}{R \vdash \sigma(a, a) = \tau(a, a)}\,\epsilon \qquad \dfrac{\dfrac{}{R \vdash a = b}\,a \quad \dfrac{}{R \vdash a = a}\,r}{R \vdash \tau(a, a) = \tau(b, a)}\,c}{R \vdash \sigma(a, b) = \tau(b, a)}\,t \times 2}{}\,b$$

---

[8] This has the effect of identifying terms like $(0, 1, 2)(y, y) =_\epsilon (0, 3)(y)$ or $(0, 1, -1)(x, x) = (0)$ by summing the weights to repeated listed variables since there can only be one copy of a transition from a state to another.

[9] We should note that, as usual in this paper, we implicitly take the semantics of states in $\mathcal{W}$-coalgebras in the final coalgebra for this functor. However, weighted automata are more often considered with weighted languages or power series as semantics.

[10] This observation is due to the upcoming book of Adámek, Milius, and Moss, but is easy enough to check by hand.

## 7. Conclusion and future directions

In this paper, we presented a sequent-style deduction system for reasoning about behavioural equivalence of points in coalgebras and specifications of finitary functors in Set. This system was based on a relaxed version of $H_\Sigma$-bisimulations which nicely coincide with $F$-bisimulations called bisimulations up to presentation. We demonstrated this proof system was sound and complete for finitary functors preserving weak pullbacks, and also sound for finitary functors without the weak pullback requirement. We also demonstrated that three common operations on functors have uniform effects on both the signature and equations in a presentation.

One restriction we would like to remove is the totality restriction. Since our specifications are defined with a (total) function, there is exactly one related coalgebra and each variable has exactly one interpretation in a final coalgebra. This makes strong soundness and completeness results decidedly less satisfying—either the assumptions are true or false of the single model of the specification. By removing the totality restriction, we could get more meaningful strong soundness and completeness.

Another more practical advantage of partial specifications is the ability to detect equality even in circumstances where values of certain variables are not known or are irrelevant. For example, from

$$d(x) = 1{:}\mathrm{zip}(y, z) \qquad d(y) = 1{:}\mathrm{zip}(x, z)$$

we would like to be able to conclude $\models x = y$ even if the value of $z$ is not specified.

We implicitly used the fact that finitary functors have *flat* presentations. That is, each point in $FX$ is an image of a point in $H_\Sigma X$ and all equations necessary for the presentation are between flat terms. Flat signatures are not always the most natural though, sometimes one would like to use zip terms like $0{:}\mathrm{zip}(1 : x, \mathrm{zip}(y, y))$, where we have three function symbols: $\Sigma_1 = \{0 :, 1 :\}$ and $\Sigma_2 = \{\mathrm{zip}\}$. Then some non-flat equations are necessary to capture all the truths of the system. How many more modifications are necessary to deal successfully with specifications in $X \to T_\Sigma X$ instead of $X \to H_\Sigma X$?

We are also interested in contexts beyond Set, particularly Vect. Milius [6] extended the expression calculi of Bonsangue et al. [4,5] to vector space coalgebras of the functor $FX = \mathbb{R} \times X$, providing a sound and complete system for reasoning about stream circuits. We have some hope that by combining our approach with the general definition of signature from Kelly and Power [21] we might be able to devise a system usable in more categories than Set.

There are also a couple of generic questions suggested by this work. Here we utilized the quotient relationship $\epsilon : H_\Sigma \twoheadrightarrow F$ to relate $H_\Sigma$- and $F$-bisimulations. Do other relationships between functors yield interesting interplay between their coalgebras? Additionally, many of the results here suggest a connection to bisimulation up to literature. How does bisimulation up to presentation fit into the theory of enhanced bisimulations?

## Acknowledgements

I owe thanks to Alexandra Silva for helpful conversations at WoLLIC at the start of this project and to Larry Moss for his expertise and encouragement throughout. Thanks also to both sets of referees for their careful reading and thoughtful comments, particularly regarding the final section.

## References

[1] Jiří Adámek, Stefan Milius, Lawrence S. Moss, On finitary functors and their presentations, in: Coalgebraic Methods in Computer Science, Springer, 2012, pp. 51–70.
[2] Lawrence S. Moss, Recursion and corecursion have the same equational logic, Theor. Comput. Sci. 294 (1) (2003) 233–267.
[3] Lawrence S. Moss, Erik Wennstrom, Glen T. Whitney, A complete logical system for the equality of recursive terms for sets, in: Logic and Program Semantics, Springer, 2012, pp. 180–203.
[4] Marcello Bonsangue, Jan Rutten, Alexandra Silva, A Kleene theorem for polynomial coalgebras, in: Foundations of Software Science and Computational Structures, Springer, 2009, pp. 122–136.
[5] Marcello Bonsangue, Jan Rutten, Alexandra Silva, An algebra for Kripke polynomial coalgebras, in: 24th Annual IEEE Symposium on Logic in Computer Science, LICS'09, IEEE, 2009, pp. 49–58.
[6] Stefan Milius, A sound and complete calculus for finite stream circuits, in: 25th Annual IEEE Symposium on Logic in Computer Science, LICS, IEEE, 2010, pp. 421–430.
[7] Alexandra M. Silva, Kleene Coalgebra, PhD thesis, CWI, 2010.
[8] Alexandra M. Silva, Marcello Maria Bonsangue, Jan J.M.M. Rutten, Kleene Coalgebras, CWI, Software Engineering [SEN], 2010.
[9] Jan JMM Rutten, Universal coalgebra: a theory of systems, Theor. Comput. Sci. 249 (1) (2000) 3–80.
[10] David Sprunger, A complete logic for behavioural equivalence in coalgebras of finitary set functors, in: International Workshop on Coalgebraic Methods in Computer Science, Springer, 2016, pp. 156–173.
[11] Saunders Mac Lane, Categories for the Working Mathematician, vol. 5, Springer Science & Business Media, 1978.
[12] Peter Aczel, Nax Mendler, A Final Coalgebra Theorem, Springer, Berlin, Heidelberg, 1989, pp. 357–365.
[13] H. Peter Gumm, Tobias Schröder, Monoid-labeled transition systems, Electron. Notes Theor. Comput. Sci. 44 (1) (2001) 185–204.
[14] Clemens Kupke, Jan J.M.M. Rutten, On the Final Coalgebra of Automatic Sequences, in: Logic and Program Semantics, Springer, 2012, pp. 149–164.
[15] Clemens Grabmayer, Jorg Endrullis, Dimitri Hendriks, Jan Willem Klop, Lawrence S. Moss, Automatic sequences and zip-specifications, in: Proceedings of the 27th Annual IEEE/ACM Symposium on Logic in Computer Science, IEEE Computer Society, 2012, pp. 335–344.
[16] Jurriaan Rot, Filippo Bonchi, Marcello Bonsangue, Damien Pous, J.J.M.M. Rutten, Alexandra Silva, Enhanced coalgebraic bisimulation, Math. Struct. Comput. Sci. (2015) 1–29.
[17] Jirí Adámek, H. Peter Gumm, Vera Trnková, Presentation of set functors: a coalgebraic perspective, J. Log. Comput. 20 (5) (2010) 991–1015.

[18] Antonius J.C. Hurkens, Monica McArthur, Yiannis N. Moschovakis, Lawrence S. Moss, Glen T. Whitney, The logic of recursive equations, J. Symb. Log. 63 (02) (1998) 451–478.
[19] Jiří Adámek, Stefan Milius, Terminal coalgebras and free iterative theories, Inf. Comput. 204 (7) (2006) 1139–1172.
[20] Filippo Bonchi, Marcello Bonsangue, Michele Boreale, Jan Rutten, Alexandra Silva, A coalgebraic perspective on linear weighted automata, Inf. Comput. 211 (2012) 77–105.
[21] G. Maxwell Kelly, A. John Power, Adjunctions whose counits are coequalizers, and presentations of finitary enriched monads, J. Pure Appl. Algebra 89 (1) (1993) 163–179.