Companions, Codensity and Causality

Damien Pous[1(✉)] and Jurriaan Rot[2,3]

[1] Univ Lyon, CNRS, ENS de Lyon, UCB Lyon 1, LIP, Lyon, France
`Damien.Pous@ens-lyon.fr`
[2] Radboud University, Nijmegen, The Netherlands
[3] CWI, Amsterdam, The Netherlands
`jrot@cs.ru.nl`

Abstract. In the context of abstract coinduction in complete lattices, the notion of compatible function makes it possible to introduce enhancements of the coinduction proof principle. The largest compatible function, called the companion, subsumes most enhancements and has been proved to enjoy many good properties. Here we move to universal coalgebra, where the corresponding notion is that of a *final* distributive law. We show that when it exists the final distributive law is a monad, and that it coincides with the codensity monad of the final sequence of the given functor. On sets, we moreover characterise this codensity monad using a new abstract notion of causality. In particular, we recover the fact that on streams, the functions definable by a distributive law or GSOS specification are precisely the causal functions. Going back to enhancements of the coinductive proof principle, we finally obtain that any causal function gives rise to a valid up-to-context technique.

1 Introduction

Coinduction has been widely studied since Milner's work on CCS [26]. In concurrency theory, it is usually exploited to define behavioural equivalences or preorders on processes and to obtain powerful proof principles. Coinduction can also be used for programming languages, to define and manipulate infinite data-structures like streams or potentially infinite trees. For instance, streams can be defined using systems of differential equations [37]. In particular, pointwise addition of two streams x, y can be defined by the following equations, where x_0 and x' respectively denote the head and the tail of the stream x.

$$\begin{aligned}(x \oplus y)_0 &= x_0 + y_0\\ (x \oplus y)' &= x' \oplus y'\end{aligned} \tag{1}$$

Coinduction as a proof principle for concurrent systems can nicely be presented at the abstract level of complete lattices [30,33]: bisimilarity is the greatest

The research leading to these results has received funding from the European Research Council (FP7/2007-2013, grant agreement nr. 320571 ; and H2020, grant agreement nr. 678157); as well as from the LABEX MILYON (ANR-10-LABX-0070, ANR-11-IDEX-0007) and the project PACE (ANR-12IS02001).

© Springer-Verlag GmbH Germany 2017
J. Esparza and A.S. Murawski (Eds.): FOSSACS 2017, LNCS 10203, pp. 106–123, 2017.
DOI: 10.1007/978-3-662-54458-7_7

fixpoint of a monotone function on the complete lattice of binary relations. In contrast, coinduction as a tool to manipulate infinite data-structures requires one more step to be presented abstractly: moving to universal coalgebra [15]. For instance, streams are the carrier of the final coalgebra of an endofunctor on Set, and simple systems of differential equations are just plain coalgebras.

In both cases one frequently needs enhancements of the coinduction principle [38,39]. Indeed, rather than working with plain bisimulations, which can be rather large, one often uses “bisimulations up-to”, which are not proper bisimulations but are nevertheless contained in bisimilarity [1,2,10,16,24,27,40]. The situation with infinite data-structures is similar. For instance, defining the shuffle product on streams is typically done using equations of the following shape,

$$\begin{aligned} (x \otimes y)_0 &= x_0 \times y_0 \\ (x \otimes y)' &= x \otimes y' \;\oplus\; x' \otimes y \end{aligned} \tag{2}$$

which fall out of the scope of plain coinduction due to the call to pointwise addition [12,37].

Enhancements of the bisimulation proof method have been introduced by Milner from the beginning [26], and further studied by Sangiorgi [38,39] and then by the first author [30,33]. Let us recall the standard formulation of coinduction in complete lattices: by Knaster-Tarski's theorem [19,42], any monotone function b on a complete lattice admits a greatest fixpoint νb that satisfies the following *coinduction principle*:

$$\frac{x \leq y \leq b(y)}{x \leq \nu b}\ \textsc{coinduction} \tag{3}$$

In words, to prove that some point x is below the greatest fixpoint, it suffices to exhibit a point y above x which is an *invariant*, i.e., a post-fixpoint of b. Enhancements, or up-to techniques, make it possible to alleviate the second requirement: instead of working with post-fixpoints of b, one might use post-fixpoints of $b \circ f$, for carefully chosen functions f:

$$\frac{x \leq y \leq b(f(y))}{x \leq \nu b}\ \textsc{coinduction up to } f \tag{4}$$

Taking inspiration from the work of Hur et al. [13], the first author recently proposed to systematically use for f the largest *compatible* function [31], i.e., the largest function t such that $t \circ b \leq b \circ t$. Such a function always exists and is called the *companion*. It enjoys many good properties, the most important one possibly being that it is a closure operator: $t \circ t = t$. Parrow and Weber also characterised it extensionally in terms of the final sequence of the function b [29,31]:

$$t : x \mapsto \bigwedge_{x \leq b_\alpha} b_\alpha \qquad \text{where } \begin{cases} b_\lambda \triangleq \bigwedge_{\alpha<\lambda} b_\alpha & \text{for limit ordinals} \\ b_{\alpha+1} \triangleq b(b_\alpha) & \text{for successor ordinals} \end{cases} \tag{5}$$

In the present paper, we give a categorical account of these ideas, generalising them from complete lattices to universal coalgebra, in order to encompass important instances of coinduction such as solving systems of equations on infinite data-structures.

Let us first be more precise about our example on streams. We consider there the Set functor $BX = \mathbb{R} \times X$, whose final coalgebra is the set $\mathbb{R}^\omega$ of streams over the reals. This means that any B-coalgebra (X, f) defines a function from X to streams. Take for instance the following coalgebra over the two-elements set $2 = \{0, 1\}$: $0 \mapsto (0.3,\ 1)$, $1 \mapsto (0.7,\ 0)$. This coalgebra can be seen as a system of two equations, whose unique solution is a function from 2 to $\mathbb{R}^\omega$, i.e., two streams, where the first has value 0.3 at all even positions and 0.7 at all odd positions.

In a similar manner, one can define binary operations on streams by considering coalgebras whose carrier consists of pairs of streams. For instance, the previous system of equations characterising pointwise addition (1) is faithfully represented by the following coalgebra:

$$\begin{aligned}(\mathbb{R}^\omega)^2 &\to B((\mathbb{R}^\omega)^2)\\ (x, y) &\mapsto (x_0 + y_0,\ \ (x', y'))\end{aligned}$$

Unfortunately, as explained above, systems of equations defining operations like shuffle product (2) cannot be represented easily in this way: we would need to call pointwise addition on streams that are not yet fully defined.

To this end, one can weaken the requirement of a B-coalgebra to that of a BF-coalgebra, when there exists a distributive law $\lambda\colon FB \Rightarrow BF$ of a monad F over B [5,12]. The proof relies on the so-called generalised powerset construction [41], and this precisely amounts to using an up-to technique. Such a use of distributive laws is actually rather standard in operational semantics [5,17,43]; they properly generalise the notion of compatible function. In order to follow [31], we thus focus on the largest distributive law.

Our first contribution consists in showing that if a functor B admits a final distributive law (called the companion), then (1) this distributive law is that of a monad T over B, and (2) any BT-coalgebra has a unique morphism to the final B-coalgebra, representing a solution to the system of equations modeled by the coalgebra (Sect. 3). In complete lattices, this corresponds to the facts that the companion is a closure operator and that it can be used as an up-to technique.

Then we move to conditions under which the companion exists. We start from the *final sequence* of the functor B, which is commonly used to obtain the existence of a final coalgebra [3,4], and we show that the companion actually coincides with the *codensity monad* of this sequence, provided that this codensity monad exists and is preserved by B (Theorem 5.1). Those conditions are satisfied by all polynomial functors. This link with the final sequence of the functor makes it possible to recover Parrow and Weber's characterisation (Eq. (5)).

We can go even further for ω-continuous endofunctors on Set: the codensity monad of the final sequence can be characterised in terms of a new abstract notion of *causal algebra* (Definition 6.1). On streams, this notion coincides with the standard notion of causality [12]: causal algebras (on streams) correspond to operations such that the n-th value of the result only depends on the n-th first values of the arguments. For instance, pointwise addition and shuffle product are causal algebras for the functor $SX = X^2$.

These two characterisations of the companion in terms of the codensity monad and in terms of causal algebras are the key theorems of the present paper. We study some of their consequences in Sect. 7.

First, given a causal algebra for a functor F, we get that any system of equations represented as a BF-coalgebra admits a unique solution. Such a technique makes it possible to define shuffle product in a streamlined way, without using distributive laws: using pointwise stream addition as a causal S-algebra, Eq. (2) can be represented by the following BS-coalgebra:

$$\begin{aligned}(\mathbb{R}^\omega)^2 &\to BS((\mathbb{R}^\omega)^2)\\ (x,y) &\mapsto (x_0 \times y_0,\ \ ((x,y')\,,\,(x',y)))\end{aligned}$$

(Intuitively, the inner pairs (x, y') and (x', y) correspond to the corecursive calls, and thus to the shuffle products $x \otimes y'$ and $x' \otimes y$; in contrast, the intermediate pair $((x, y')\,,\,(x', y))$ corresponds to a call to the causal algebra on S, i.e., in this case, pointwise addition.) In the very same way, with the functor $BX = 2 \times X^A$ for deterministic automata, we immediately obtain the semantics of non-deterministic automata and context-free grammars using simple causal algebras on formal languages (Examples 7.1 and 7.2).

Second, we obtain that algebras on the final coalgebra are causal if and only if they can be defined by a distributive law. Similar results were known to hold for streams [12] and languages [35]. Our characterisation is more abstract and less syntactic; the precise relationship between those results remains to be studied.

Third, we can combine our results with some recent work [6] where we rely on (bi)fibrations to lift distributive laws on systems (e.g., automata, LTSs) to obtain up-to techniques for coinductive predicates or relations on those systems (e.g., language equivalence, bisimilarity, divergence). Doing so, we obtain that every causal algebra gives rise to a valid up-to context technique (Sect. 7.3). For instance, bisimulation up to pointwise additions and shuffle products is a valid technique for proving stream equalities coinductively.

We conclude with an expressivity result (Sect. 8): while abstract GSOS specifications [43] seem more expressive than plain distributive laws, we show that this is actually not the case: any algebra obtained from an abstract GSOS specification can actually be defined from a plain distributive law.

2 Preliminaries

A *coalgebra* for a functor $B\colon \mathcal{C} \to \mathcal{C}$ is a pair (X, f) where X is an object in $\mathcal{C}$ and $f\colon X \to BX$ a morphism. A coalgebra homomorphism from (X, f) to (Y, g) is a $\mathcal{C}$-morphism $h\colon X \to Y$ such that $g \circ h = Fh \circ f$. A coalgebra (Z, ζ) is called *final* if it is final in the category of coalgebras , i.e., for every coalgebra (X, f) there exists a unique coalgebra morphism from (X, f) to (Z, ζ).

An *algebra* for a functor $F\colon \mathcal{D} \to \mathcal{D}$ is defined dually to a coalgebra, i.e., it is a pair (X, a) where $a\colon FX \to X$, and an algebra morphism from (X, a) to (Y, b) is a morphism $h\colon X \to Y$ such that $h \circ a = b \circ Fh$.

A *monad* is a triple (T, η, μ) where $T\colon \mathcal{C} \to \mathcal{C}$ is a functor, and $\eta\colon \mathsf{Id} \Rightarrow T$ and $\mu\colon TT \Rightarrow T$ are natural transformations called *unit* and *multiplication* respectively, such that $\mu \circ T\eta = \mathsf{id} = \mu \circ \eta T$ and $\mu \circ \mu T = \mu \circ T\mu$.

Distributive Laws. A *distributive law* of a functor $F\colon \mathcal{C} \to \mathcal{C}$ over a functor $B\colon \mathcal{C} \to \mathcal{C}$ is a natural transformation $\lambda\colon FB \Rightarrow BF$. If B has a final coalgebra (Z, ζ), then such a λ induces a unique algebra α making the following commute.

$$\begin{tikzcd}
FZ \arrow[r, "F\zeta"] \arrow[d, "\alpha"'] & FBZ \arrow[r, "\lambda_Z"] & BFZ \arrow[d, "B\alpha"] \\
Z \arrow[rr, "\zeta"'] & & BZ
\end{tikzcd}$$

We call α the *algebra induced by* λ (on the final coalgebra).

Let (T, η, μ) be a monad. A distributive law of (T, η, μ) over B is a natural transformation $\lambda\colon TB \Rightarrow BT$ such that $B\eta = \lambda \circ \eta B$ and $\lambda \circ \mu B = B\mu \circ \lambda T \circ T\lambda$.

Final Sequence. Let $B\colon \mathcal{C} \to \mathcal{C}$ be an endofunctor on a complete category $\mathcal{C}$. The *final sequence* is the unique ordinal-indexed sequence defined by $B_0 = 1$ (the final object of $\mathcal{C}$), $B_{i+1} = BB_i$ and $B_j = \lim_{i<j} B_i$ for a limit ordinal j, with connecting morphisms $B_{j,i}\colon B_j \to B_i$ for all $i \leq j$, satisfying $B_{i,i} = \mathsf{id}$, $B_{j+1,i+1} = BB_{j,i}$ and if j is a limit ordinal then $(B_{j,i})_{i<j}$ is a limit cone.

The final sequence is a standard tool for constructing final coalgebras: if there exists an ordinal k such that $B_{k+1,k}$ is an isomorphism, then $B^{-1}_{k+1,k}\colon B_k \to BB_k$ is a final B-coalgebra [4, Theorem 1.3] (and dually for initial algebras [3]). In the sequel, we shall sometimes present it as a functor $\bar{B}\colon \mathsf{Ord}^{\mathsf{op}} \to \mathcal{C}$, given by $\bar{B}(i) = B_i$ and $\bar{B}(j, i) = B_{j,i}$.

Example 2.1. Consider the functor $B\colon \mathsf{Set} \to \mathsf{Set}$ given by $BX = A \times X$, whose coalgebras are stream systems. Then $B_0 = 1$ and $B_{i+1} = A \times B_i$ for $0 < i < \omega$. Hence, for $i < \omega$, B_i is the set of all finite lists over A of length i. The limit B_ω consists of the set of all streams over A. For each i, j with $i \leq j$, the connecting map $B_{j,i}$ maps a stream (if $j = \omega$) or a list (if $j < \omega$) to the prefix of length i. The set B_ω of streams is a final B-coalgebra.

Example 2.2. For the Set functor $BX = 2 \times X^A$ whose coalgebras are deterministic automata over A, B_i is (isomorphic to) the set of languages of words over A with length below i. In particular, $B_\omega = \mathcal{P}(A^*)$ is the set of all languages, and it is a final B-coalgebra.

A functor $B\colon \mathcal{C} \to \mathcal{C}$ is called (ω)-*continuous* if it preserves limits of ω^{op}-chains. For such a functor, B_ω is the carrier of a final B-coalgebra. The functors of stream systems and automata in the above examples are both ω-continuous.

3 Properties of the Companion

Definition 3.1. *Let $B\colon \mathcal{C} \to \mathcal{C}$ be a functor. The category $\mathsf{DL}(B)$ of distributive laws is defined as follows. An object is a pair (F, λ) where $F\colon \mathcal{C} \to \mathcal{C}$ is a functor and $\lambda\colon FB \Rightarrow BF$ is a natural transformation. A morphism from (F, λ) to (G, ρ) is a natural transformation $\kappa\colon F \Rightarrow G$ s.t. $\rho \circ \kappa B = B\kappa \circ \lambda$. The* companion *of B is the final object of $\mathsf{DL}(B)$, if it exists.*

$$\begin{tikzcd} FB \arrow[r, Rightarrow, "\kappa B"] \arrow[d, Rightarrow, "\lambda"'] & GB \arrow[d, Rightarrow, "\rho"] \\ BF \arrow[r, Rightarrow, "B\kappa"'] & BG \end{tikzcd}$$

Morphisms in $\mathsf{DL}(B)$ are a special case of *morphisms of distributive laws*, see [18, 22, 34, 44]. In the remainder of this section, we assume that the companion of B exists, and we denote it by (T, τ). We first prove that it is a monad.

Theorem 3.1. *There are unique $\eta\colon \mathsf{Id} \Rightarrow T$ and $\mu\colon TT \Rightarrow T$ such that (T, η, μ) is a monad and $\tau\colon TB \Rightarrow BT$ is a distributive law of this monad over B.*

Proof. Define η and μ as the unique morphisms from id_B and $\tau T \circ T\tau$ respectively to the companion:

$$\begin{tikzcd} B \arrow[r, equal] \arrow[d, Rightarrow, "\eta B"'] & B \arrow[d, Rightarrow, "B\eta"] \\ TB \arrow[r, Rightarrow, "\tau"] & BT \end{tikzcd} \qquad \begin{tikzcd} TTB \arrow[r, Rightarrow, "T\tau"] \arrow[d, Rightarrow, "\mu B"'] & TBT \arrow[r, Rightarrow, "\tau T"] & BTT \arrow[d, Rightarrow, "B\mu"] \\ TB \arrow[rr, Rightarrow, "\tau"] & & BT \end{tikzcd}$$

By definition, they satisfy the required axioms for τ to be a distributive law of monad over functor. The proof that (T, η, μ) is indeed a monad is routine, using finality of (T, τ), see the appendix [32]. □

A distributive law λ of a monad over a functor allows one to strengthen the coinduction principle obtained by finality, as observed in [5] (specifically its Corollary 4.3.6), where it is called λ-*coiteration*. This principle allows one to solve (co)recursive equations, see, e.g., loc. cit. and [14, 25]. Since the companion is a distributive law of a monad (Theorem 3.1) we obtain the following.

Corollary 3.1. *Let (Z, ζ) be a final B-coalgebra. For every morphism $f\colon X \to BTX$ there is a unique morphism $f^\dagger\colon X \to Z$ such that the following commutes:*

$$\begin{tikzcd} X \arrow[rr, "f^\dagger"] \arrow[d, "f"'] & & Z \arrow[d, "\zeta"] \\ BTX \arrow[r, "BTf^\dagger"'] & BTZ \arrow[r, "B\alpha"'] & BZ \end{tikzcd}$$

where α is the algebra induced by the distributive law τ of the companion.

Instantiated to the complete lattice case, this is a soundness result: any invariant up to the companion (a post-fixpoint of $b \circ t$) is below the greatest fixpoint (νb).

Now assume that $\mathcal{C}$ has an initial object 0. One can define the final coalgebra and the algebra induced by the companion explicitly:

Theorem 3.2. *The B-coalgebra $(T0, \tau_0 \circ T!_{B0})$ is final, and the algebra induced on it by the companion is given by μ_0.*

More generally, the algebra induced by any distributive law factors through the algebra μ_0 induced by the companion.

Proposition 3.1. *Let (T, η, μ) be the monad on the companion (Theorem 3.1). Let $\lambda\colon FB \Rightarrow BF$ be a distributive law, and $\alpha\colon FT0 \Rightarrow T0$ the algebra on the final coalgebra induced by it. Let $\bar{\lambda}\colon F \Rightarrow T$ be the unique natural transformation induced by finality of the companion. Then $\alpha = \mu_0 \circ \bar{\lambda}_{T0}$.*

4 The Codensity Monad

The notion of *codensity monad* is a special instance of a right Kan extension, which plays a central role in the following sections. We briefly define them here; see [20,21,28] for a comprehensive study.

$$\begin{tikzcd} HF \arrow[rr, Rightarrow, "\kappa F"] \arrow[dr, Rightarrow, "\alpha"'] & & IF \arrow[dl, Rightarrow, "\beta"] \\ & G & \end{tikzcd}$$

Given $F\colon \mathcal{C} \to \mathcal{D}$, $G\colon \mathcal{C} \to \mathcal{E}$ be two functors. Define the category $\mathcal{K}(F, G)$ whose objects are pairs (H, α) of a functor $H\colon \mathcal{D} \to \mathcal{E}$ and a natural transformation $\alpha\colon HF \Rightarrow G$. A morphism from (H, α) to (I, β) is a natural transformation $\kappa\colon H \Rightarrow I$ such that $\beta \circ \kappa F = \alpha$.

The *right Kan extension* of G along F is a final object $(\mathsf{Ran}_F G, \epsilon)$ in $\mathcal{K}(F, G)$; the natural transformation $\epsilon\colon (\mathsf{Ran}_F G)F \Rightarrow G$ is called its *counit*. A functor $K\colon \mathcal{E} \to \mathcal{F}$ is said to *preserve* $\mathsf{Ran}_F G$ if $K \circ \mathsf{Ran}_F G$ is a right Kan extension of KG along F, with counit $K\epsilon\colon K(\mathsf{Ran}_F G)F \Rightarrow KG$.

$$\begin{tikzcd} HF \arrow[rr, Rightarrow, "\hat{\alpha}F"] \arrow[dr, Rightarrow, "\alpha"'] & & \mathsf{C}_F F \arrow[dl, Rightarrow, "\epsilon"] \\ & F & \end{tikzcd}$$

The codensity monad is a special case, with $F = G$. Explicitly, the *codensity monad* of a functor $F\colon \mathcal{C} \to \mathcal{D}$ consists of a functor $\mathsf{C}_F\colon \mathcal{D} \to \mathcal{D}$ and a natural transformation $\epsilon\colon \mathsf{C}_F F \Rightarrow F$ s.t. for every functor $H\colon \mathcal{D} \to \mathcal{D}$ and natural transformation $\alpha\colon HF \Rightarrow F$ there is a unique $\hat{\alpha}\colon H \Rightarrow \mathsf{C}_F$ s.t. $\epsilon \circ \hat{\alpha}F = \alpha$.

As the name suggests, C_F is a monad: the unit η and the multiplication μ are the unique natural transformations such that $\epsilon \circ \eta F = \mathsf{id}$ and $\epsilon \circ \mu F = \epsilon \circ \mathsf{C}_F \epsilon$. In the sequel we will abbreviate the category $\mathcal{K}(F, F)$ as $\mathcal{K}(F)$.

Right Kan extensions can be computed pointwise as a limit, if sufficient limits exist. For an object X in $\mathcal{D}$, denote by $\Delta_X\colon \mathcal{C} \to \mathcal{D}$ the functor that maps every object to X. By Δ_X/F we denote the comma category, where an object is a pair (Y, f) consisting of an object Y in $\mathcal{C}$ and an arrow $f\colon X \to FY$ in $\mathcal{D}$, and an arrow from (Y, f) to (Z, g) is a map $h\colon Y \to Z$ in $\mathcal{C}$ such that $Fh \circ f = g$. There is a forgetful functor $(\Delta_X/F) \to \mathcal{C}$, which remains unnamed below.

Lemma 4.1. *Let $F\colon \mathcal{C} \to \mathcal{D}$, $G\colon \mathcal{C} \to \mathcal{E}$ be functors. If, for every object X in $\mathcal{D}$, the limit $\lim \Big((\Delta_X/F) \to \mathcal{C} \xrightarrow{G} \mathcal{D} \Big)$ exists, then the right Kan extension $\mathsf{Ran}_F G$ exists, and is given on an object X by that limit.*

The codensity monad of a functor F is the right Kan extension of F along itself. Hence, Lemma 4.1 gives us a way of computing the codensity monad.

The hypotheses are met in particular if $\mathcal{C}$ is essentially small (equivalent to a category with a set of objects and a set of arrows) and $\mathcal{D}$ is locally small and complete. The latter conditions hold for $\mathcal{D} = \mathsf{Set}$. In that case, we have the following concrete presentation; see, e.g., [8, Sect. 2.5] for a proof.

Lemma 4.2. *Let $F\colon \mathcal{C} \to \mathsf{Set}$ be a functor, where $\mathcal{C}$ is essentially small. The codensity monad C_F is given by $\mathsf{C}_F(X) = \{\alpha\colon (F-)^X \Rightarrow F\}$ and, for $h\colon X \to Y$, $(\mathsf{C}_F(h)(\alpha))_A\colon (FA)^Y \to FA$ is given by $f \mapsto \alpha_A(f \circ h)$. The natural transformation $\epsilon\colon \mathsf{C}_F F \Rightarrow F$ is given by $\epsilon_X(\alpha\colon F^{FX} \Rightarrow F) = \alpha_X(\mathsf{id}_{FX})$.*

5 Constructing the Companion by Codensity

It is standard in the theory of coalgebras to compute the final coalgebra of a functor B as a limit of the final sequence $\bar{B}$, see Sect. 2. In this section, we focus on the codensity monad of the final sequence, and show that it yields—under certain conditions—the companion of B.

The codensity monad of $\bar{B}$ is final in the category of natural transformations of the form $F\bar{B} \Rightarrow \bar{B}$ (see Sect. 4), whereas the companion of B is final in the category of distributive laws over B. The following lemma is a first step towards connecting companion and codensity monad.

Lemma 5.1. *For every $\lambda\colon FB \Rightarrow BF$ there exists a unique $\alpha\colon F\bar{B} \Rightarrow \bar{B}$ such that for all $i \in \mathsf{Ord}$: $\alpha_{i+1} = B\alpha_i \circ \lambda_{B_i}$. Moreover, if $B_{k+1,k}$ is an isomorphism for some k, then α_k is the algebra induced by λ on the final coalgebra.*

We turn to the main result of this section: the codensity monad of $\bar{B}$ yields the companion of B, if B preserves this codensity monad. The latter condition, as well as the concrete form of the companion computed in this manner, becomes clearer when we instantiate this result to the case where $\mathcal{C}$ is a lattice (Sect. 5.1) and the case $\mathcal{C} = \mathsf{Set}$ (Sect. 6).

Theorem 5.1. *Let $\bar{B}\colon \mathsf{Ord}^{\mathsf{op}} \to \mathcal{C}$ be the final sequence of an endofunctor B. If the codensity monad $\mathsf{C}_{\bar{B}}$ exists and B preserves it (as a right Kan extension) then there is a distributive law τ of the codensity monad $(\mathsf{C}_{\bar{B}}, \eta, \mu)$ over B such that $(\mathsf{C}_{\bar{B}}, \tau)$ is the companion of B.*

Proof (Outline). The preservation assumption means that $(B\mathsf{C}_{\bar{B}}, B\epsilon)$ is a right Kan extension of $B\bar{B}$ along $\bar{B}$. The natural transformation τ is defined, using the universal property of $B\epsilon$, as the unique $\tau\colon \mathsf{C}_{\bar{B}}B \Rightarrow B\mathsf{C}_{\bar{B}}$ such that $B\epsilon_i \circ \tau_{B_i} = \epsilon_{i+1}\colon \mathsf{C}_{\bar{B}}BB_i \Rightarrow BB_i$ for all i. See the appendix for a full proof [33]. $\square$

The following result characterises the algebra induced on the final coalgebra by the distributive law of the companion, in terms of the counit ϵ of the codensity monad of $\bar{B}$. This plays an important role for the case $\mathcal{C} = \mathsf{Set}$ (Sect. 7).

Proposition 5.1. *Suppose B is a functor satisfying the hypotheses of Theorem 5.1. Let $(\mathsf{C}_{\bar{B}}, \epsilon)$ be the codensity monad of $\bar{B}$, with distributive law τ and monad structure $(\mathsf{C}_{\bar{B}}, \eta, \mu)$. If $B_{k+1,k}$ is an isomorphism for some k, then*

1. *$\epsilon_k\colon \mathsf{C}_{\bar{B}}B_k \to B_k$ is the algebra induced by τ on the final coalgebra;*
2. *if $\mathcal{C}$ has an initial object 0 then ϵ_k is isomorphic to μ_0.*

5.1 Codensity and the Companion of a Monotone Function

Throughout this section, let $b\colon L \to L$ be a monotone function on a complete lattice. By Theorem 5.1, the companion of a monotone function b (viewed as a functor on a poset category) is given by the right Kan extension of the final sequence $\bar{b}\colon \mathsf{Ord}^{\mathsf{op}} \to L$ along itself. Using Lemma 4.1, we obtain the characterisation of the companion given in the Introduction (5).

Theorem 5.2. *The companion t of b is given by*

$$t : x \mapsto \bigwedge_{x \leq b_i} b_i$$

Proof. By Lemma 4.1, the codensity monad $\mathsf{C}_{\bar{b}}$ can be computed by

$$\mathsf{C}_{\bar{b}}(x) = (\mathsf{Ran}_{\bar{b}}\bar{b})(x) = \bigwedge_{x \leq b_i} b_i \,,$$

a limit that exists since L is a complete lattice. We apply Theorem 5.1 to show that $\mathsf{C}_{\bar{b}}$ is the companion of b. The preservation condition of the theorem amounts to the equality $b \circ \mathsf{Ran}_{\bar{b}}\bar{b} = \mathsf{Ran}_{\bar{b}}(b \circ \bar{b})$ which, by Lemma 4.1, in turn amounts to

$$b(\bigwedge_{x \leq b_i} b_i) = \bigwedge_{x \leq b_i} b(b_i)$$

for all $x \in L$. The sequence $(b_i)_{i \in \mathsf{Ord}}$ is decreasing and stagnates at some ordinal ϵ; therefore, the two intersections collapse into their last terms, say b_δ and $b(b_\delta)$ (with δ the greatest ordinal such that $x \not\leq b_{\delta+1}$, or ϵ if such an ordinal does not exist). The equality follows. $\square$

In fact, the category $\mathcal{K}(b)$ defined in Sect. 4 instantiates to the following: an object is a monotone function $f\colon L \to L$ such that $f(b_i) \leq b_i$ for all $i \in \mathsf{Ord}$, and an arrow from f to g exists iff $f \leq g$. The companion t is final in this category. This yields the following characterisation of functions below the companion.

Proposition 5.2. *Let t be the companion of b. For any monotone function f we have $f \leq t$ iff $\forall i \in \mathsf{Ord} : f(b_i) \leq b_i$.*

A key intuition about up-to techniques is that they should at least preserve the greatest fixpoint (i.e., up-to context is valid only when bisimilarity is a congruence). It is however well-known that this is not a sufficient condition [38,39]. The above proposition gives a stronger and better intuition: a technique should preserve all approximations of the greatest fixpoint (the elements of the final sequence) to be below the companion, and thus sound.

This intuition on complete lattices leads us to the abstract notion of causality we introduce in the following section.

6 Causality by Codensity

We focus on the codensity monad of the final sequence of an ω-continuous Set endofunctor B. For such a functor, B_ω is the carrier of a final coalgebra and Lemma 4.2 provides us with a description of the codensity monad in terms of natural transformations of the form $(\bar{B}-)^X \Rightarrow \bar{B}$. We show that such natural transformations correspond to a new abstract notion which we call *causal algebras*. Based on this correspondence and Theorem 5.1, we will get a concrete understanding of the companion of B in Sect. 7.

Definition 6.1. *Let $B, F\colon \mathsf{Set} \to \mathsf{Set}$ be functors. An algebra $\alpha\colon FB_\omega \to B_\omega$ is called (ω)-*causal *if for every set X, functions $f, g\colon X \to B_\omega$ and $i < \omega$:*

$$\begin{tikzcd} & B_\omega \arrow[dr, "B_{\omega,i}"] & \\ X \arrow[ur, "f"] \arrow[dr, "g"'] & & B_i \\ & B_\omega \arrow[ur, "B_{\omega,i}"'] & \end{tikzcd} \quad \textit{implies} \quad \begin{tikzcd} & FB_\omega \arrow[r, "\alpha"] & B_\omega \arrow[dr, "B_{\omega,i}"] & \\ FX \arrow[ur, "Ff"] \arrow[dr, "Fg"'] & & & B_i \\ & FB_\omega \arrow[r, "\alpha"'] & B_\omega \arrow[ur, "B_{\omega,i}"'] & \end{tikzcd}$$

Causal algebras form a category $\mathsf{causal}(B)$: an object is a pair $(F, \alpha\colon FB_\omega \to B_\omega)$ where α is causal, and a morphism from (F, α) to (G, β) is a natural transformation $\kappa\colon F \Rightarrow G$ such that $\beta \circ \kappa_{B_\omega} = \alpha$.

*An (ω)-*causal function *on $|V|$ arguments is a causal algebra for the functor $(-)^V$. Equivalently, $\alpha\colon (B_\omega)^V \to B_\omega$ is causal iff for every $h, k \in (B_\omega)^V$ and every $i < \omega$: if $B_{\omega,i} \circ h = B_{\omega,i} \circ k$ then $B_{\omega,i} \circ \alpha(h) = B_{\omega,i} \circ \alpha(k)$.*

Example 6.1. Recall from Example 2.1 that, for the functor $BX = A \times X$, B_i is the set of lists of length i, and in particular B_ω is the set of streams over A. We focus first on causal *functions*. To this end, for $\sigma, \tau \in B_\omega$, we write $\sigma \equiv_i \tau$ if σ and τ are equal up to i, i.e., $\sigma(k) = \tau(k)$ for all $k < i$. It is easy to verify that a function of the form $\alpha\colon (B_\omega)^n \to B_\omega$ is causal iff for all $\sigma_1, \ldots, \sigma_n, \tau_1, \ldots, \tau_n$ and all $i < \omega$: if $\sigma_j \equiv_i \tau_j$ for all $j \leq n$ then $\alpha(\sigma_1, \ldots, \sigma_n) \equiv_i \alpha(\tau_1, \ldots, \tau_n)$.

For instance, taking $n = 2$, $\mathsf{alt}(\sigma, \tau) = (\sigma(0), \tau(1), \sigma(2), \tau(3), \ldots)$ is causal, whereas $\mathsf{even}(\sigma) = (\sigma(0), \sigma(2), \ldots)$ (with $n = 1$) is not causal. For $A = \mathbb{R}$, standard operations from the stream calculus such as pointwise stream addition, shuffle product and shuffle product are all causal.

The above notion of causal functions (with a finite set of arguments V) agrees with the standard notion of causal stream functions (e.g., [12]). Our notion of causal *algebras* generalises it from single functions to algebras for arbitrary functors. This includes polynomial functors modelling a signature. For $A = \mathbb{R}$, the algebra $\alpha\colon \mathcal{P}_\omega(B_\omega) \to B_\omega$ for the finite powerset functor $\mathcal{P}_\omega$, defined by $\alpha(S)(n) = \min\{\sigma(n) \mid \sigma \in S\}$ is a causal algebra which is not a causal function. The algebra $\beta\colon \mathcal{P}_\omega(B_\omega) \to B_\omega$ given by $\beta(S)(n) = \sum_{\sigma \in S} \sigma(n)$ is *not* causal according to Definition 6.1. Intuitively, $\beta(\{\sigma, \tau\})(i)$ depends on equality of σ and τ, since addition of real numbers is not idempotent.

Example 6.2. For the functor $BX = 2 \times X^A$, $B_\omega = \mathcal{P}(A^*)$ is the set of languages over A (Example 2.2). Given languages L and K, we write $L \equiv_i K$ if L and K

contain the same words of length below i. A function $\alpha\colon (\mathcal{P}(A^*))^n \to \mathcal{P}(A^*)$ is causal iff for all languages $L_1, \dots, L_n, K_1, \dots, K_n$: if $L_j \equiv_i K_j$ for all $j \le n$ then $\alpha(L_1, \dots, L_n) \equiv_i \alpha(K_1, \dots, K_n)$. For instance, union, concatenation, Kleene star, and shuffle of languages are all causal. An example of a causal algebra that is not a causal function is $\alpha\colon \mathcal{P}(\mathcal{P}(A^*)) \to \mathcal{P}(A^*)$ defined by union.

The following result connects causal algebras to natural transformations of the form $F\bar{B} \Rightarrow \bar{B}$ (which, from Sect. 4, form a category $\mathcal{K}(\bar{B})$).

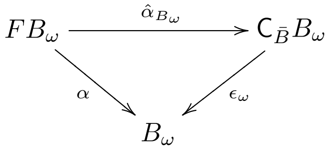
Theorem 6.1. *Let $B, F\colon \mathsf{Set} \to \mathsf{Set}$ be functors, and suppose B is ω-continuous. The category $\mathsf{causal}(B)$ of causal algebras is isomorphic to the category $\mathcal{K}(\bar{B})$. Concretely, there is a one-to-one correspondence*

$$\frac{\alpha\colon F\bar{B} \Rightarrow \bar{B}}{\alpha_\omega\colon FB_\omega \to B_\omega \ \textit{causal}}$$

From top to bottom, this is given by evaluation at ω. Moreover, we have $\beta \circ \kappa\bar{B} = \alpha$ iff $\beta_\omega \circ \kappa_{B_\omega} = \alpha_\omega$ for any $\alpha\colon F\bar{B} \Rightarrow \bar{B}$, $\beta\colon G\bar{B} \Rightarrow \bar{B}$ and $\kappa\colon F \Rightarrow G$.

By the above theorem, the universal property of the codensity monad amounts to the following property of causal algebras.

Corollary 6.1. *Suppose $B\colon \mathsf{Set} \to \mathsf{Set}$ is ω-continuous. Let ϵ be the counit of $\mathsf{C}_{\bar{B}}$. Then ϵ_ω is final in $\mathsf{causal}(B)$, i.e., for every causal algebra $\alpha\colon FB_\omega \to B_\omega$, there is a unique natural transformation $\hat{\alpha}\colon F \Rightarrow \mathsf{C}_{\bar{B}}$ such that $\epsilon_\omega \circ \hat{\alpha}_{B_\omega} = \alpha$.*



$$\begin{tikzcd}
FB_\omega \arrow[rr, "\hat{\alpha}_{B_\omega}"] \arrow[dr, "\alpha"'] & & \mathsf{C}_{\bar{B}}B_\omega \arrow[dl, "\epsilon_\omega"] \\
& B_\omega &
\end{tikzcd}$$

By Lemmas 4.2 and 6.1, we obtain the following concrete description of the codensity monad $\mathsf{C}_{\bar{B}}$ of the final sequence of a Set endofunctor B, as a *functor of causal functions.*

Theorem 6.2. *Let $B\colon \mathsf{Set} \to \mathsf{Set}$ be an ω-continuous functor. The codensity monad $\mathsf{C}_{\bar{B}}$ of the final sequence of B is given by*

$$\mathsf{C}_{\bar{B}}(X) = \{\alpha\colon B_\omega^X \to B_\omega \mid \alpha \textit{ is a causal function}\},$$
$$\mathsf{C}_{\bar{B}}(h\colon X \to Y)(\alpha) = \lambda f.\, \alpha(f \circ h),$$

and, for the counit $\epsilon\colon \mathsf{C}_{\bar{B}}\bar{B} \Rightarrow \bar{B}$, we have $\epsilon_\omega(\alpha\colon B_\omega^{B\omega} \to B_\omega) = \alpha(\mathsf{id}_{B_\omega})$.

Hence, the codensity monad of the final sequence of the functor $X \mapsto A \times X$ of stream systems maps a set X to the set of all causal stream functions with $|X|$ arguments. Similarly for the functor $X \mapsto 2 \times X^A$: we obtain a functor of causal functions on languages.

7 Companion of a Set Functor

The previous sections gives us a concrete understanding of the codensity monad of the final sequence of a Set functor in terms of causal functions, and Theorem 5.1 provides us with a sufficient condition for this codensity monad to be the companion. We now focus on several applications of these results.

A rather general class of functors that satisfy the hypotheses of Theorem 5.1 is given by the *polynomial functors*. Automata, stream systems, Mealy and Moore machines, various kinds of trees, and many more are all examples of coalgebras for polynomial functors (e.g., [15]). A functor $B\colon \mathsf{Set} \to \mathsf{Set}$ is called polynomial (in a single variable) if it is isomorphic to a functor of the form

$$X \mapsto \coprod_{a \in A} X^{B_a}$$

for some A-indexed collection $(B_a)_{a \in A}$ of sets. As explained in [11, 1.18], a Set functor B is polynomial if and only if it preserves connected limits. This implies existence and preservation by B of the codensity monad of $\bar{B}$, as required by Theorem 5.1 (see the appendix for details [32]).

Lemma 7.1. *If $B\colon \mathsf{Set} \to \mathsf{Set}$ is polynomial, then it satisfies the hypotheses of Theorem 5.1.*

As a consequence, if B is polynomial, the functor of causal functions in Theorem 6.2 is the companion of B.

7.1 Solving Equations via Causal Algebras

As explained in the introduction, a distributive law of F over B allows one to solve systems of equations, formalised in terms of BF-coalgebras, leading to an expressive coinductive definition technique. This approach is formally supported by a solution theorem, stated for the companion in Corollary 3.1. Based on the characterisation of the companion in terms of causal algebras, we obtain a new, simplified solution theorem: it does not mention distributive laws at all, but is stated purely in terms of causal algebras.

Theorem 7.1. *Let $B\colon \mathsf{Set} \to \mathsf{Set}$ be a polynomial functor, with final coalgebra (B_ω, ζ). Let $\alpha\colon FB_\omega \to B_\omega$ be a causal algebra. For every $f\colon X \to BFX$ there is a unique $f^\dagger\colon X \to Z$ such that the following diagram commutes.*

$$\begin{tikzcd}
X \arrow[rr, "f^\dagger"] \arrow[d, "f"'] & & B_\omega \arrow[d, "\zeta"] \\
BFX \arrow[r, "BFf^\dagger"'] & BFB_\omega \arrow[r, "B\alpha"'] & BB_\omega
\end{tikzcd}$$

Example 7.1. For the functor $BX = A \times X$, B_ω is the set of streams. Take $SX = X^2$ for F, and consider the coalgebra $f\colon 1 \to BS1$ with $1 = \{*\}$, defined by $* \mapsto (1, (*, *))$. Pointwise addition is a causal function on streams, modelled by an algebra on B_ω for the functor S. By Theorem 7.1 we obtain a unique solution $\sigma \in B_\omega$, satisfying $\sigma_0 = 1$ and $\sigma' = \sigma \oplus \sigma$. Similarly, the shuffle product of streams is causal, so that by applying Theorem 7.1 with that algebra and the same coalgebra f we obtain a unique stream σ satisfying $\sigma_0 = 1$, $\sigma' = \sigma \otimes \sigma$.

As explained in the Introduction, this method also allows one to define functions on streams. For instance, for the shuffle product, define a BS-coalgebra $f\colon (B_\omega)^2 \to BS(B_\omega)^2$, by $f(\sigma, \tau) = (\sigma_0 \times \tau_0, ((\sigma', \tau), (\tau, \sigma')))$. Since addition of streams is causal, by Theorem 7.1 there is a unique $f^\dagger\colon B_\omega \times B_\omega \to B_\omega$ such that $f^\dagger(\sigma, \tau)(0) = \sigma(0) \times \tau(0)$ and $(f^\dagger(\sigma, \tau))' = (f^\dagger(\sigma', \tau) \oplus f^\dagger(\sigma, \tau'))$, matching the definition given in the Introduction (2). Notice that not every function defined in this way is causal; for instance, it is easy to define even (see Example 6.1), even with the standard coinduction principle (i.e., where $F = \mathsf{Id}$ and $\alpha = \mathsf{id}$).

Example 7.2. Consider the functor $BX = 2 \times X^A$, whose final coalgebra consists of the set $\mathcal{P}(A^*)$ of languages. A $B\mathcal{P}$-coalgebra $f\colon X \to 2 \times (\mathcal{P}(X))^A$ is a non-deterministic automaton. Taking the causal algebra $\alpha\colon \mathcal{P}(\mathcal{P}(A^*)) \to \mathcal{P}(A^*)$ defined by union, the unique map $f^\dagger\colon X \to \mathcal{P}(A^*)$ from Theorem 7.1 is the usual language semantics of non-deterministic automata.

In [45], a context-free grammar (in Greibach normal form) is modelled as a $B\mathcal{P}^*$-coalgebra $f\colon X \to 2 \times (\mathcal{P}(X)^*)^A$, and its semantics is defined operationally by turning f into a deterministic automaton over $\mathcal{P}(X^*)$. In [36] this operational view is related to the semantics of CFGs in terms of language equations. Consider the causal algebra $\alpha\colon \mathcal{P}(\mathcal{P}(A^*)^*) \to \mathcal{P}(A^*)$ defined by union and language composition: $\alpha(S) = \bigcup_{L_1 \ldots L_k \in S} L_1 L_2 \ldots L_k$. By Theorem 7.1, any context-free grammar f has a unique solution in languages, which is the semantics of CFGs in the usual sense. As such, we obtain an elementary coalgebraic semantics of CFGs that does not require us to relate it to an operational semantics.

7.2 Causal Algebras and Distributive Laws

Another application of the fact that the codensity monad is the companion is that the final causal algebra in Corollary 6.1 is, by Proposition 5.1, the algebra induced by a distributive law. Hence, *any* causal algebra is "definable" by a distributive law, in the sense that it factors as a (component of a) natural transformation followed by the algebra induced by a distributive law.

More precisely, suppose $B\colon \mathsf{Set} \to \mathsf{Set}$ has a final coalgebra (Z, ζ). We say an algebra $\alpha\colon FZ \to Z$ is *definable by a distributive law over* B if there exists a distributive law $\lambda\colon GB \Rightarrow BG$ with induced algebra $\beta\colon GZ \to Z$ and a natural transformation $\kappa\colon F \Rightarrow G$ such that the following commutes:

$$\begin{tikzcd}
FZ \arrow[rr, "\kappa_Z"] \arrow[dr, "\alpha"'] & & GZ \arrow[dl, "\beta"] \\
& Z &
\end{tikzcd}$$

Theorem 7.2. *Let $B \colon \mathsf{Set} \to \mathsf{Set}$ be polynomial. An algebra $\alpha \colon FB_\omega \to B_\omega$ is causal if and only if it is definable by a distributive law over B.*

Since the functors for stream systems and automata are polynomial, as a special case of Theorem 7.2 we obtain that a stream function, or a function on languages, is causal if and only if it is definable by a distributive law.

In [12], a similar result is shown concretely for causal stream functions, and this is extended to languages in [35]. In both cases, very specific presentations of distributive laws for the systems at hand are used to present the distributive law based on a "syntax", which however is not too clearly distinguished from the semantics: it consists of a single operation symbol for every causal function. In our case, in the proof of Theorem 7.2, we use the *companion*, which consists of the actual functions rather than a syntactic representation. Indeed, the setting of Theorem 7.2 applies more abstractly to *all* causal algebras, not just causal functions. However, it remains an intriguing question how to obtain a concrete syntactic characterisation of a distributive law for a given causal algebra.

7.3 Soundness of Up-to Techniques

The *contextual closure* of an algebra is one of the most powerful up-to techniques, which allows one to exploit algebraic structure in bisimulation proofs. In [7], it is shown that the contextual closure is sound (compatible) on any bialgebra for a distributive law. Here, we move away from distributive laws and give an elementary condition for soundness of the contextual closure on the final coalgebra: that the algebra under consideration is causal. In fact, we prove that this implies that the contextual closure lies below the companion, which not only gives soundness, but also allows to combine it with other up-to techniques.

Due to space limitations, we can not fully explain the relevant definitions, and refer to [7] for details. Bisimulations on a B-coalgebra (X, f) are the post-fixed points of a monotone function $b_f \colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ on the lattice Rel_X of relations on X, defined by $b_f(R) = f^* \circ \mathsf{Rel}(B)(R)$. Here $\mathsf{Rel}(B)$ is the *relation lifting* of B, and f^* is inverse image along $f \times f$, see, e.g., [15]. Contextual closure $\mathsf{ctx}_\alpha \colon \mathsf{Rel}_X \to \mathsf{Rel}_X$ with respect to an algebra $\alpha \colon FX \to X$ is defined dually by $\mathsf{ctx}_\alpha(R) = \coprod_\alpha \circ \mathsf{Rel}(F)(R)$, where $\coprod_\alpha$ is direct image along $\alpha \times \alpha$.

Theorem 7.3. *Let $B \colon \mathsf{Set} \to \mathsf{Set}$ be polynomial, and (B_ω, ζ) a final B-coalgebra. Let t_ζ be the companion of b_ζ. For any causal algebra $\alpha \colon FB_\omega \to B_\omega$: $\mathsf{ctx}_\alpha \leq t_\zeta$.*

This implies that one can safely use the contextual closure for *any* causal algebra, such as union, concatenation and Kleene star of languages, or product and sum of streams. Endrullis et al. [9] prove the soundness of *causal contexts* in combination with other up-to techniques, for equality of streams. The soundness of causal algebras for streams is a special case of Theorem 7.3, but the latter provides more: being below the companion, it is possible to compose it to other such functions to obtain combined up-to techniques in a modular fashion, cf. [31].

8 Abstract GSOS

To obtain expressive specification formats, Turi and Plotkin [43] use natural transformations of the form $\lambda\colon F(B \times \mathsf{Id}) \Rightarrow BF^*$, where F^* is the free monad for F. These are the so-called *abstract GSOS specifications.* We conclude this article by showing that they are actually equally expressive as plain distributive laws of a functor F over B.

If B has a final coalgebra (Z, ζ), then any abstract GSOS specification $\lambda\colon F(B \times \mathsf{Id}) \Rightarrow BF^*$ defines an algebra $\alpha\colon FZ \to Z$ on it, which is the unique algebra making the following diagram commute.

$$\begin{tikzcd}
FZ \arrow[r, "F\langle\zeta,\mathsf{id}\rangle"] \arrow[d, "\alpha"'] & F(B \times \mathsf{Id})Z \arrow[r, "\lambda_Z"] & BF^*Z \arrow[d, "B\alpha^*"] \\
Z \arrow[rr, "\zeta"'] & & BZ
\end{tikzcd}$$

Here α^* is the Eilenberg-Moore algebra for the free monad corresponding to α. Intuitively, this algebra gives the interpretation of the operations defined by λ.

Like plain distributive laws (Lemma 5.1), abstract GSOS specifications induce natural transformations of the form $F\bar{B} \Rightarrow \bar{B}$.

Lemma 8.1. *For every $\lambda\colon F(B \times \mathsf{Id}) \Rightarrow BF^*$ there is a unique $\alpha\colon F\bar{B} \Rightarrow \bar{B}$ such that for all $i \in \mathsf{Ord}$: $\alpha_{i+1} = B\alpha_i^* \circ \lambda_{B_i} \circ F\langle \mathsf{id}, B_{i+1,i}\rangle$. Moreover, if $B_{k+1,k}$ is an isomorphism for some k, then α_k is the algebra induced by λ on the final coalgebra.*

This places abstract GSOS specifications within the framework of the companion, constructed via the codensity monad of the final sequence $\bar{B}$. Whenever that construction applies (e.g., for polynomial functors), any algebra defined by an abstract GSOS is thus already definable by a plain distributive law over B.

Theorem 8.1. *Suppose $B\colon \mathcal{C} \to \mathcal{C}$ satisfies the conditions of Theorem 5.1. Every algebra induced on the final coalgebra by an abstract GSOS specification $\lambda\colon F(B \times \mathsf{Id}) \Rightarrow BF^*$ is definable by a distributive law over B (cf. Sect. 7.2).*

In this sense, abstract GSOS is no more expressive than plain distributive laws. Note, however, that this does involve moving to a different (larger) syntax.

Remark 8.1. Every abstract GSOS specification $\lambda\colon F(B \times \mathsf{Id}) \Rightarrow BF^*$ corresponds to a unique distributive law $\lambda^\dagger\colon F^*(B \times \mathsf{Id}) \Rightarrow (B \times \mathsf{Id})F^*$ of the free monad F^* over the (cofree) copointed functor $B \times \mathsf{Id}$, see [23]. The algebra induced by λ decomposes as the algebra induced by $\lambda^\dagger$ and the canonical natural transformation $F \Rightarrow F^*$. This implies that every algebra induced by an abstract GSOS is definable by a distributive law over the copointed functor $B \times \mathsf{Id}$. Theorem 8.1 strengthens this to definability by a distributive law over B.

Acknowledgments. We are grateful to Henning Basold, Filippo Bonchi, Bart Jacobs, Joshua Moerman, Daniela Petrişan, and Jan Rutten for valuable discussions and comments.

References

1. Abadi, M., Gordon, A.D.: A bisimulation method for cryptographic protocols. Nord. J. Comput. **5**(4), 267 (1998)
2. Abramsky, S.: The lazy lambda calculus. In: Research Topics in Functional Programming, pp. 65–116. Addison Wesley (1990)
3. Adámek, J.: Free algebras and automata realizations in the language of categories. Commentationes Mathematicae Universitatis Carolinae **15**(4), 589–602 (1974)
4. Barr, M.: Algebraically compact functors. J. Pure Appl. Algebra **82**(3), 211–231 (1992)
5. Bartels, F.: On generalised coinduction and probabilistic specification formats. PhD thesis, CWI, Amsterdam, April 2004
6. Bonchi, F., Petrisan, D., Pous, D., Rot, J.: Coinduction up-to in a fibrational setting. In: Proceeding CSL-LICS, pp. 20:1–20:9. ACM (2014)
7. Bonchi, F., Petrişan, D., Pous, D., Rot, J.: A general account of coinduction up-to. Acta Informatica, pp. 1–64 (2016)
8. Cordier, J.-M., Porter, T.: Shape Theory: Categorical Methods of Approximation. Mathematics and its Applications. Ellis Horwood, New York (1989). Reprinted by Dover, 2008
9. Endrullis, J., Hendriks, D., Bodin, M.: Circular coinduction in coq using bisimulation-up-to techniques. In: Blazy, S., Paulin-Mohring, C., Pichardie, D. (eds.) ITP 2013. LNCS, vol. 7998, pp. 354–369. Springer, Heidelberg (2013). doi:10.1007/978-3-642-39634-2_26
10. Fournet, C., Lévy, J.-J., Schmitt, A.: An asynchronous, distributed implementation of mobile ambients. In: Leeuwen, J., Watanabe, O., Hagiya, M., Mosses, P.D., Ito, T. (eds.) TCS 2000. LNCS, vol. 1872, pp. 348–364. Springer, Heidelberg (2000). doi:10.1007/3-540-44929-9_26
11. Gambino, N., Kock, J.: Polynomial functors and polynomial monads. In: Proceeding of Mathematical proceedings of the cambridge philosophical society, vol. 154, pp. 153–192. Cambridge University Press (2013)
12. Hansen, H.H., Kupke, C., Rutten, J.: Stream differential equations: specification formats and solution methods. CoRR, abs/1609.08367 (2016)
13. Hur, C., Neis, G., Dreyer, D., Vafeiadis, V.: The power of parameterization in coinductive proof. In: Proceeding of POPL, pp. 193–206. ACM (2013)
14. Jacobs, B.: Distributive laws for the coinductive solution of recursive equations. Inf. Comput. **204**(4), 561–587 (2006)
15. Jacobs, B.: Introduction to coalgebra. Towards mathematics of states and observations. Draft (2014)
16. Jeffrey, A., Rathke, J.: A theory of bisimulation for a fragment of concurrent ML with local names. Theor. Comput. Sci. **323**(1–3), 1–48 (2004)
17. Klin, B.: Bialgebras for structural operational semantics: an introduction. Theor. Comput. Sci. **412**(38), 5043–5069 (2011)
18. Klin, B., Nachyla, B.: Presenting morphisms of distributive laws. In: Proceeding CALCO, vol. 35. LIPIcs, pp. 190–204. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2015)

19. Knaster, B.: Un théorème sur les fonctions d'ensembles. Annales de la Socit Polonaise de Mathmatiques **6**, 133–134 (1928)
20. Lane, S.M.: Categories for the Working Mathematician. Springer, New York (1998)
21. Leinster, T.: Codensity and the ultrafilter monad. Theory and Applications of Categories **28**(13), 332–370 (2013)
22. Lenisa, M., Power, J., Watanabe, H.: Distributivity for endofunctors, pointed and co-pointed endofunctors, monads and comonads. Electron. Notes Theor. Comput. Sci. **33**, 230–260 (2000)
23. Lenisa, M., Power, J., Watanabe, H.: Category theory for operational semantics. Theor. Comput. Sci. **327**(1–2), 135–154 (2004)
24. Leroy, X.: Formal verification of a realistic compiler. Commun. ACM **52**(7), 107–115 (2009)
25. Milius, S., Moss, L.S., Schwencke, D.: Abstract GSOS rules and a modular treatment of recursive definitions. Logical Methods Comput. Sci. **9**(3) (2013)
26. Milner, R.: Communication and Concurrency. Prentice Hall (1989)
27. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes I/II. Inf. Comput. **100**(1), 1–77 (1992)
28. nLab article. Kan extension 2016. (Revision 103), http://ncatlab.org/nlab/show/Kan+extension
29. Parrow, J., Weber, T.: The largest respectful function. Logical Methods Comput. Sci. **12**(2) (2016)
30. Pous, D.: Complete lattices and up-to techniques. In: Shao, Z. (ed.) APLAS 2007. LNCS, vol. 4807, pp. 351–366. Springer, Heidelberg (2007). doi:10.1007/978-3-540-76637-7_24
31. Pous,D.: Coinduction all the way up. In: Proceeding LICS, pp. 307-316. ACM (2016)
32. Pous, D., Rot, J.: Companions, Codensity, and Causality. In: Esparza, J., Murawski, A.S. (eds.) FOSSACS 2017 LNCS, vol. 10203, pp. 106–123. Springer, Heidelberg (2017). (version with proofs), https://hal.archives-ouvertes.fr/hal-01442222
33. Pous, D., Sangiorgi, D.: Advanced Topics in Bisimulation and Coinduction, chapter about "Enhancements of the coinductive proof method". Cambridge University Press (2011)
34. Power, J., Watanabe, H.: Combining a monad and a comonad. Theor. Comput. Sci. **280**(1–2), 137–162 (2002)
35. Rot, J., Bonsangue, M.M., Rutten, J.: Proving language inclusion and equivalence by coinduction. Inf. Comput. **246**, 62–76 (2016)
36. Rot, J., Winter, J.: On language equations and grammar coalgebras for context-free languages. In: Proceeding CALCO Early Ideas (2013)
37. Rutten, J.J.M.M.: A coinductive calculus of streams. Math. Struct. Comput. Sci. **15**(1), 93–147 (2005)
38. Sangiorgi, D.: On the bisimulation proof method. Math. Struct. Comput. Sci. **8**, 447–479 (1998)
39. Sangiorgi, D., Walker, D.: The π-calculus: a theory of mobile processes. Cambridge University Press (2001)
40. Sevcík, J., Vafeiadis, V., Nardelli, F.Z., Jagannathan, S., Sewell, P.: CompCertTSO: a verified compiler for relaxed-memory concurrency. J. ACM **60**(3), 22 (2013)
41. Silva, A., Bonchi, F., Bonsangue, M., Rutten, J.: Generalizing the powerset construction, coalgebraically. In: Proceeding FSTTCS, pp. 272–283 (2010)
42. Tarski, A.: A lattice-theoretical fixpoint theorem, its applications. Pacific J. Math. **5**(2), 285–309 (1955)

43. Turi, D., Plotkin, G.D.: Towards a mathematical operational semantics. In: Proceeding LICS, pp. 280-291. IEEE (1997)
44. Watanabe, H.: Well-behaved translations between structural operational semantics. Electron. Notes Theor. Comput. Sci. **65**(1), 337–357 (2002)
45. Winter, J., Bonsangue, M.M., Rutten, J.J.M.M.: Coalgebraic characterizations of context-free languages. Logical Methods Comput. Sci. **9**(3) (2013)