PRESENTING DISTRIBUTIVE LAWS

MARCELLO M. BONSANGUE, HELLE H. HANSEN, ALEXANDER KURZ, AND JURRIAAN ROT

LIACS – Leiden University, Netherlands, and Formal Methods – Centrum Wiskunde & Informatica, Amsterdam, Netherlands

ESS/TBM – Delft University of Technology, Netherlands, and Formal Methods – Centrum Wiskunde & Informatica, Amsterdam, Netherlands

Dep. of Computer Science – University of Leicester, United Kingdom

LIACS – Leiden University, Netherlands, and Formal Methods – Centrum Wiskunde & Informatica, Amsterdam, Netherlands

Abstract. Distributive laws of a monad $\mathcal{T}$ over a functor F are categorical tools for specifying algebra-coalgebra interaction. They proved to be important for solving systems of corecursive equations, for the specification of well-behaved structural operational semantics and, more recently, also for enhancements of the bisimulation proof method. If $\mathcal{T}$ is a free monad, then such distributive laws correspond to simple natural transformations. However, when $\mathcal{T}$ is not free it can be rather difficult to prove the defining axioms of a distributive law. In this paper we describe how to obtain a distributive law for a monad with an equational presentation from a distributive law for the underlying free monad. We apply this result to show the equivalence between two different representations of context-free languages.

1. Introduction

The combination of algebraic structure and observable behaviour is fundamental in computer science. Examples include the operational models of structural operational semantics [3], denotational models of programming languages [37], finite stream circuits [24], linear and context-free systems of behavioural differential equations [30, 38], and many types of automata such as nondeterministic and weighted automata [32].

In the categorical treatment of these examples, the algebraic structure is encoded by a monad $\mathcal{T} = \langle T, \eta, \mu \rangle$, and the system behaviour by coalgebras for a functor F. Often it is desirable that the algebraic and coalgebraic structure are compatible in some way. A

1998 ACM Subject Classification: D.3.1, F.1.1, F.3.0, F.3.2.

Key words and phrases: Coalgebra, algebra, distributive laws, abstract GSOS, monad, equational presentation.

The research of Helle H. Hansen has been funded by the Netherlands Organisation for Scientific Research (NWO), Veni project number 639.021.231.

The research of Jurriaan Rot has been funded by the Netherlands Organisation for Scientific Research (NWO), CoRE project, dossier number: 612.063.920.

LOGICAL METHODS IN COMPUTER SCIENCE DOI:10.2168/LMCS-???

general approach to specifying such algebra-coalgebra interaction is via a distributive law. There are several advantages of this structured approach. A distributive law λ of the monad $\mathcal{T}$ over F induces a $\mathcal{T}$-algebra on the final F-coalgebra of behaviours, yields solutions to corecursive equations $\phi\colon X \to FTX$ [7], and ensures that bisimulation is a congruence [34]. Moreover, it yields the soundness of techniques such as bisimulation-up-to-context [7] and extensions thereof [27, 29].

Describing a distributive law explicitly and proving that it is one can, however, be rather complicated. Therefore, general methods for constructing distributive laws from simpler ingredients are very useful. An important example of this is given by abstract GSOS [34, 7, 21] where distributive laws of a free monad $\mathcal{T}$ over a (copointed) functor F are shown to correspond to plain natural transformations, called *abstract GSOS-rules* as they can be seen as specification formats. In [12] it was shown how an abstract GSOS-rule for a free monad $\mathcal{T}$ and functor F can be lifted to one for the functor $F(-)^A$ which describes F-systems with input in A. Another method which works for all monads $\mathcal{T}$, but only for certain polynomial behaviour functors F, produces a distributive law inducing a "pointwise lifting" of $\mathcal{T}$-algebra structure to F-behaviours, cf. [13, 14, 32].

But many examples do not fit into the abovementioned settings. An important motivating example for this paper is that of context-free grammars, where sequential composition is not a pointwise operation and whose formal semantics satisfies the axioms of idempotent semirings, i.e., the algebraic structure is not free. More generally, one may be interested in a monad arising from a free one by adding equations which one knows to hold in the final coalgebra.

The main contribution of this paper is to give a general approach for constructing a distributive law λ' for a monad $\mathcal{T}'$ with an equational presentation, from a distributive law λ for the underlying free monad $\mathcal{T}$. We have no constraints on the behaviour functor F. This λ' is obtained as a certain quotient of λ by the equations $\mathcal{E}$ of $\mathcal{T}'$, hence we say that λ' *is presented by a* λ *for the free monad and the equations* $\mathcal{E}$. We show that such quotients exist precisely when the distributive law *preserves the equations* $\mathcal{E}$, which roughly means that congruences generated by the equations are bisimulations. We also discuss how these quotients of distributive laws give rise to quotients of bialgebras, thereby giving a concrete operational interpretation, and a correspondence between solutions to corecursive equations with and without equations. As an illustration and application of our theory, we show the existence of a distributive law of the monad for idempotent semirings over the deterministic automata functor. This result yields the equivalence betweeen the Greibach normal form representation of context-free languages and the coalgebraic representation via context-free expressions given in [38].

Outline. In Section 2, we recall the notions of monads and algebras, and give a concrete description of monad quotients. In Section 3, we recall distributive laws and their application to solving systems of equations. Then, in Section 4, we prove our main results on quotients of distributive laws. In Section 5, we show that such quotients give rise to quotients of bialgebras. Finally, in Section 6, we discuss related work, and provide some directions for future work.

This paper is an extended version of [9]. It contains all proofs and generalises the main results of [9] from monads on the category of Set to monads on arbitrary categories (with the appropriate structure in the category of algebras). Furthermore, this paper includes the treatment of distributive laws of a monad over a comonad, allowing more general specification formats that involve look-ahead in the premises.

Acknowledgements. We thank Neil Ghani, Bart Jacobs, Jan Rutten, Jiří Velebil and Joost Winter for helpful discussions and suggestions. We are indebted to the referees for their constructive comments, which greatly helped us improving the paper.

2. Monads, Algebras and Equations

We start by recalling some basic definitions on monads, algebras, term equations and congruences (see, e.g., [6, 4] for a detailed introduction). We will then proceed to give a concrete description of the quotient monad arising from a free monad and a set of equations.

Let C be a category. A *monad* is a triple $\mathcal{T} = \langle T, \eta, \mu \rangle$ where T is an endofunctor on C, and $\eta\colon \mathrm{Id} \Rightarrow T$ and $\mu\colon TT \Rightarrow T$ are natural transformations such that $\mu \circ T\eta = \mathrm{id} = \mu \circ \eta_T$ and $\mu \circ \mu_T = \mu \circ T\mu$. A *$\mathcal{T}$-algebra* is a pair $\langle A, \alpha \rangle$ where A is a C-object and $\alpha\colon TA \to A$ is an arrow such that $\alpha \circ \eta_A = \mathrm{id}$ and $\alpha \circ \mu_A = \alpha \circ T\alpha$. A *($\mathcal{T}$-algebra) homomorphism* from $\langle A, \alpha \rangle$ to $\langle B, \beta \rangle$ is an arrow $f\colon A \to B$ such that $f \circ \alpha = \beta \circ Tf$. The *free $\mathcal{T}$-algebra* over a C-object X is $\langle TX, \mu_X \rangle$. Given any $\mathcal{T}$-algebra $\langle A, \alpha \rangle$ and any arrow $f\colon X \to A$, there is a unique algebra homomorphism $f^\sharp\colon TX \to A$ such that $f^\sharp \circ \eta_X = f$, given by $f^\sharp = \alpha \circ Tf$. We denote the category of $\mathcal{T}$-algebras and their homomorphisms by $\mathsf{Alg}(\mathcal{T})$, and the associated forgetful functor by $U\colon \mathsf{Alg}(\mathcal{T}) \to \mathsf{C}$.

Let $\langle T, \eta, \mu \rangle$ and $\langle K, \theta, \nu \rangle$ be monads. A *morphism of monads* is a natural transformation $\sigma\colon T \Rightarrow K$ such that the following diagram commutes:

$$\begin{array}{ccccc} \mathrm{Id} & \xrightarrow{\eta} & T & \xleftarrow{\mu} & TT \\ & \searrow^{\theta} & \downarrow{\scriptstyle \sigma} & & \downarrow{\scriptstyle \sigma\sigma} \\ & & K & \xleftarrow{\nu} & KK \end{array} \tag{2.1}$$

where $\sigma\sigma = K\sigma \circ \sigma_T = \sigma_K \circ T\sigma$.

Assume we are given a monad $\mathcal{T} = \langle T, \eta, \mu \rangle$ on some category C. We define *$\mathcal{T}$-equations* as a 3-tuple $\mathcal{E} = \langle E, l, r \rangle$ where E is an endofunctor on C and $l, r\colon E \Rightarrow T$ are natural transformations. The intuition is that E models the arity of the equations, and l and r give the left and right-hand side, respectively. This is illustrated below by an example.

Example 2.1. Consider the Set functor $\Sigma X = X \times X + 1$, modelling a binary operation and a constant, which we call $+$ and 0 respectively. The (underlying functor of the) free monad T_Σ for Σ sends a set X to the terms over X built from $+$ and 0. The equations $x + 0 = x$, $x + y = y + x$ and $(x + y) + z = x + (y + z)$ can be modelled as follows. The functor E is defined as $EX = X + (X \times X) + (X \times X \times X)$. The natural transformations $l, r\colon E \Rightarrow T_\Sigma$ are given by $l_X(x) = x + 0$ and $r_X(x) = x$ for all $x \in X$; $l_X(x, y) = x + y$ and $r_X(x, y) = y + x$ for all $(x, y) \in X \times X$; $l_X(x, y, z) = x + (y + z)$ and $r_X(x, y, z) = (x + y) + z$ for all $(x, y, z) \in X \times X \times X$.

Throughout this paper we will need assumptions on C, $\mathcal{T}$, and $\mathcal{E}$. For this section we only need the following.

Assumption 2.2. We assume that $\mathcal{T}$ is a monad on C, and $E\colon \mathsf{C} \to \mathsf{C}$ is a functor such that:

(1) $\mathsf{Alg}(\mathcal{T})$ has coequalizers.
(2) U and TU map regular epis in $\mathsf{Alg}(\mathcal{T})$ to epis in C.
(3) EU maps regular epis in $\mathsf{Alg}(\mathcal{T})$ to epis in C.

The first condition is needed to construct quotients of free algebras modulo equations. The second condition relates quotients of algebras (regular epis) with quotients in the base category (epis). It is satisfied if either U preserves regular epis or if U preserves epis. Finally, the last condition is satisfied if E preserves epimorphisms in C.

Example 2.3.

(1) If $\mathsf{C} = \mathsf{Set}$ the conditions are satisfied for any monad $\mathcal{T}$ and endofunctor E.
(2) If C is the category Pos of posets with monotone maps the second condition may fail, see Example 6 in [8], which can be adapted to show that the monad induced by the adjunction $\mathbb{2} \times (-) \dashv (-)^{\mathbb{2}} \colon \mathsf{Pos} \to \mathsf{Pos}$ does map some regular epis to non-epis.
(3) If C is abelian groups and T is the identity, the well-known fact that torsion-free abelian groups are not monadic over Set [10] can be adapted to give an example of equations E that fail condition 3. Let $\mathbb{Z}_n$ denote the group of integers with addition modulo n. Define $E : \mathsf{C} \to \mathsf{C}$ as the coproduct of abelian groups $E(A) = \coprod_{n\in\mathbb{N}} \mathsf{C}(\mathbb{Z}_n, A)$ and define $l, r \colon E(A) \to A$ by $l(g) = g(1)$ and $r(g) = 0$. Note that there is $g \colon \mathbb{Z}_n \to A$ only if $n \cdot g(1) = 0$, that is, only if $g(1)$ 'has torsion'. The equations then force the quotient of A to be torsion-free (i.e., no element different than 0 has torsion). Since $E(\mathbb{Z}) = 1$ and $E(\mathbb{Z}_2)$ is infinite, the epi $\mathbb{Z} \to \mathbb{Z}_2$ is not preserved. $\mathsf{Alg}(T, \mathcal{E})$ is the category of torsion-free abelian groups.

Let $\mathbb{A} = \langle A, \alpha\rangle$ be a $\mathcal{T}$-algebra. We denote by $s_{\mathbb{A}}$ the coequalizer of $l^\sharp_A \circ \alpha$ and $r^\sharp_A \circ \alpha$ in $\mathsf{Alg}(\mathcal{T})$ depicted in the following diagram (we suppress the forgetful functor and denote by the same symbol both an algebra morphism and its underlying C-arrow)

$$\langle TEA, \mu_{EA}\rangle \underset{r^\sharp_A}{\overset{l^\sharp_A}{\rightrightarrows}} \langle TA, \alpha\rangle \xrightarrow{\ \alpha\ } \langle A, \alpha\rangle \xrightarrow{\ s_{\mathbb{A}}\ } \langle A/\mathcal{E}, \alpha_{\mathcal{E}}\rangle\,. \tag{2.2}$$

In the case $\mathsf{C} = \mathsf{Set}$, this entails that $\ker(s_{\mathbb{A}})$ is the *congruence* generated by the set $E_{\mathbb{A}} = \{(\alpha(l_A(e)), \alpha(r_A(e)) \mid e \in EA\}$, i.e., by the least equivalence relation on A that includes $E_{\mathbb{A}}$ and is a subalgebra of $\langle A, \alpha\rangle \times \langle A, \alpha\rangle$. In this sense, the kernel pair of a morphism always yields a congruence, and conversely, every congruence relation on an algebra $\langle A, \alpha\rangle$ is the kernel of the corresponding quotient homomorphism. In general, the coequaliser (2.2) in $\mathsf{Alg}(\mathcal{T})$ differs from the one obtained by applying the forgetful functor U and then computing the coequaliser of $\alpha \circ l^\sharp_A$ and $\alpha \circ r^\sharp_A$ in Set. The coequalisers in $\mathsf{Alg}(T)$ and Set coincide if the equations are reflexive in the sense that the two parallel arrows $\alpha \circ l_A$ and $\alpha \circ r_A$ from EA to A have a common section, and the forgetful functor U preserves reflexive coequalisers. If T is finitary, then U preserves reflexive coequalisers. Moreover, we note that if U preserves reflexive coequalisers then T preserves them too, but not every Set-functor preserves reflexive coequalisers, cf. [5, Example 4.3].

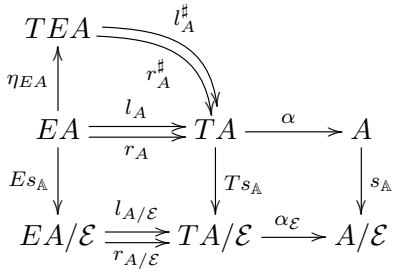
A $\mathcal{T}$-algebra $\mathbb{A} = \langle A, \alpha\rangle$ is said to *satisfy* $\mathcal{E}$ if the following diagram commutes:

$$EA \underset{r_A}{\overset{l_A}{\rightrightarrows}} TA \xrightarrow{\ \alpha\ } A\,.$$

By $\mathsf{Alg}(\mathcal{T}, \mathcal{E})$ we denote the full subcategory of $\mathcal{T}$-algebras that satisfy $\mathcal{E}$. As coequalisers are unique only up to isomorphism, we will choose s such that for all $\mathbb{A} \in \mathsf{Alg}(\mathcal{T}, \mathcal{E})$, $s_{\mathbb{A}} = \mathrm{id}_{\mathbb{A}}$.

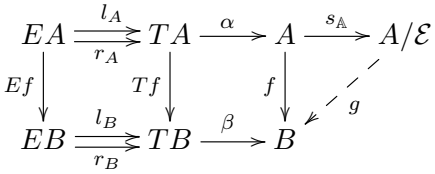
Lemma 2.4. *The inclusion $V \colon \mathsf{Alg}(\mathcal{T}, \mathcal{E}) \to \mathsf{Alg}(\mathcal{T})$ has a left-adjoint $H \colon \mathsf{Alg}(\mathcal{T}) \to \mathsf{Alg}(\mathcal{T}, \mathcal{E})$ with unit $\overline{\eta}_{\mathbb{A}} = s_{\mathbb{A}} : \langle A, \alpha\rangle \to \langle A/\mathcal{E}, \alpha_{\mathcal{E}}\rangle$ for all $\mathbb{A} \in \mathsf{Alg}(\mathcal{T})$, and counit $\overline{\epsilon}_{\mathbb{A}} = \mathrm{id}_{\mathbb{A}}$ the identity for all $\mathbb{A} \in \mathsf{Alg}(\mathcal{T}, \mathcal{E})$. In particular, $\mathsf{Alg}(\mathcal{T}, \mathcal{E})$ is a full, reflective subcategory of $\mathsf{Alg}(\mathcal{T})$.*

Proof. We first show that for any $\mathbb{A} = \langle A, \alpha\rangle$ in $\mathsf{Alg}(\mathcal{T})$, $\langle A/\mathcal{E}, \alpha_{\mathcal{E}}\rangle$ is an object in $\mathsf{Alg}(\mathcal{T}, \mathcal{E})$. Consider the following diagram:



The right-hand square commutes by the definition of $s_{\mathbb{A}}$, cf. (2.2). The left squares (for l and r respectively) commute by naturality of l and r. The upper two paths from TEA to $A/\mathcal{E}$ commute by definition of $s_{\mathbb{A}}$. From the above diagram we obtain $\alpha_{\mathcal{E}} \circ l_{A/\mathcal{E}} \circ E(s_{\mathbb{A}}) = \alpha_{\mathcal{E}} \circ r_{A/\mathcal{E}} \circ E(s_{\mathbb{A}})$ which implies $\alpha_{\mathcal{E}} \circ l_{A/\mathcal{E}} = \alpha_{\mathcal{E}} \circ r_A$ since $E(s_{\mathbb{A}})$ is epic (cf. Assumption 2.2).

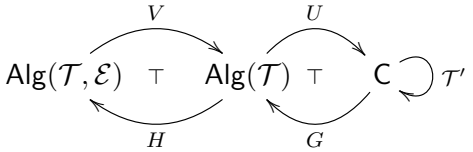
It remains to show that for $\mathbb{B} = \langle B, \beta\rangle$ in $\mathsf{Alg}(\mathcal{T}, \mathcal{E})$ and an algebra morphism $f : A \to B$ there is a unique algebra morphism $g : A/\mathcal{E} \to B$ such that $g \circ s_{\mathbb{A}} = f$. Since $\mathbb{B}$ satisfies the equations we know $\beta \circ l_B = \beta \circ r_B$, hence $f \circ \alpha \circ l_A = f \circ \alpha \circ r_A$. Since $s_{\mathbb{A}} : A \to A/\mathcal{E}$ is the coequalizer of $(\alpha \circ l_A, \alpha \circ r_A)$ the claim follows. The situation is illustrated here:



We have shown that $s_{\langle A,\alpha\rangle} \colon \langle A, \alpha\rangle \to \langle A/\mathcal{E}, \alpha_{\mathcal{E}}\rangle$ is an $\mathsf{Alg}(\mathcal{T}, \mathcal{E})$-reflection arrow for $\langle A, \alpha\rangle$. By defining $H \colon \mathsf{Alg}(\mathcal{T}) \to \mathsf{Alg}(\mathcal{T}, \mathcal{E})$ as $H\langle A, \alpha\rangle = \langle A/\mathcal{E}, \alpha_{\mathcal{E}}\rangle$, then H is left adjoint to V, and the unit of the adjunction is $\overline{\eta} = q$. Now, since the unit and counit must satisfy $V(\epsilon_{\mathbb{A}}) \circ s_{V\mathbb{A}} = id_{V\mathbb{A}}$, for all $\mathbb{A} \in \mathsf{Alg}(\mathcal{T}, \mathcal{E})$, it follows from $s_{V\mathbb{A}} = \mathrm{id}_{\mathbb{A}}$ and $VH = \mathrm{Id}_{\mathsf{Alg}(\mathcal{T},\mathcal{E})}$ that $V(\epsilon_{\mathbb{A}}) = V(id_{\mathbb{A}})$, and hence $\epsilon_{\mathbb{A}} = id_{\mathbb{A}}$. □

By composition of adjoints, the functor $UV \colon \mathsf{Alg}(\mathcal{T}, \mathcal{E}) \to \mathsf{Alg}(\mathcal{T}) \to \mathsf{C}$ has a left adjoint given by $X \mapsto \langle TX/\mathcal{E}, (\mu_X)_{\mathcal{E}}\rangle$. In what follows, we will write $T'X$ for $TX/\mathcal{E}$. This allows the following definition.

Definition 2.5 (Quotient monad)**.** Given a monad $\mathcal{T} = \langle T, \eta, \mu\rangle$ on C and $\mathcal{T}$-equations $\mathcal{E}$, we define the *quotient monad* $\mathcal{T}' = \langle T', \eta', \mu'\rangle$ as the monad on C arising from the composition of the adjunction $\langle H, V, \overline{\eta} = s, \overline{\epsilon} = \mathrm{id}\rangle$ of Lemma 2.4 and the Eilenberg-Moore adjunction $\langle G, U, \eta, \epsilon\rangle$ of $\mathcal{T}$:



We define $q \colon T \Rightarrow T'$ as the family of underlying C-arrows of reflection arrows for free algebras, i.e.,

$$q_X = Us_{\langle TX,\mu_X\rangle} \colon TX \to T'X \tag{2.3}$$

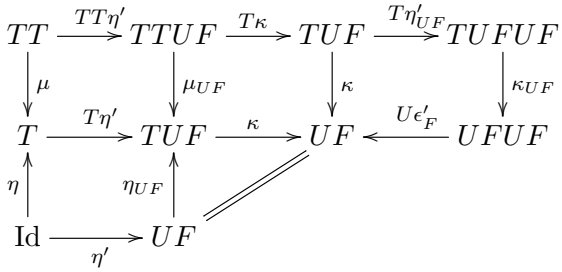
Naturality of q is clear, since s is natural. Next, we show that q is a monad morphism from $\mathcal{T}$ to $\mathcal{T}'$. One way of doing so is to show that q is a coequaliser in the category of monads and monad morphisms. Kelly studied colimits in categories of monads, and proved their existence in the context of some adjunction [17, Proposition 26.4]; with a bit of effort one can instantiate this to the adjunction constructed above. For a self-contained presentation in this section, we do not invoke Kelly's results but instead prove directly the part that shows the existence of a monad morphism. This is instantiated below to the adjunction of the quotient monad.

Lemma 2.6. *Let* A *be any subcategory of* $\mathsf{Alg}(\mathcal{T})$*, and suppose the forgetful functor* $U : \mathsf{A} \to \mathsf{C}$ *has a left adjoint* F*, with unit and counit denoted by* η' *and* ϵ' *respectively. Then*

(1) F induces a natural transformation $\kappa : TUF \Rightarrow UF$ so that $\kappa \circ T\eta' : T \Rightarrow UF$ is a monad morphism.

(2) Precomposing the functor $\mathsf{Alg}(UF) \to \mathsf{Alg}(\mathcal{T})$ induced by this monad morphism with the comparison functor $\mathsf{A} \to \mathsf{Alg}(UF)$ yields the inclusion $\mathsf{A} \to \mathsf{Alg}(\mathcal{T})$.

Proof. The functor F sends any C-object X to a T-algebra structure on UFX; we define κ_X to be that algebra structure. Naturality of κ is immediate since Ff is a $\mathcal{T}$-algebra homomorphism for any C-arrow f. To see that $\kappa \circ T\eta'$ is a monad morphism, consider:



The top left square commutes by naturality and the middle square since any component of κ is an $\mathcal{T}$-algebra. For the right square we have

$$\kappa = \kappa \circ TU\epsilon'_F \circ T\eta'_{UF} = U\epsilon'_F \circ \kappa_{UF} \circ T\eta'_{UF}$$

where the first equality follows from the triangle identity $\mathrm{id}_{UF} = U\epsilon'_F \circ \eta'_{UF}$ (and functoriality), and the second from the fact that ϵ'_{FX} is a $\mathcal{T}$-algebra homomorphism from κ_{UFX} to κ_X. The bottom left square commutes by naturality, and the triangle since κ is an $\mathcal{T}$-algebra.

For (2), we first note that the composite functor under consideration maps any T-algebra $\langle A, \alpha\rangle$ in A to $U\epsilon'_{U\langle A,\alpha\rangle} \circ \kappa_A \circ T\eta'_A$. But we have

$$\alpha = \alpha \circ TU\epsilon'_{\langle A,\alpha\rangle} \circ T\eta'_A = U\epsilon'_{U\langle A,\alpha\rangle} \circ \kappa_A \circ T\eta'_A$$

where the first equality is a triangle identity and the second is the fact that $\epsilon'_{\langle A,\alpha\rangle}$ is an algebra morphism. □

Remark 2.7. Lemma 2.6 is a special case of what is known as the structure-semantics adjointness, which establishes an adjunction between (algebraic theories or) monads over C (the structure) and 'forgetful' functors $\mathsf{A} \to \mathsf{C}$ (the semantics), see [20], [22], [33] and, in particular, [11, Theorem II.1.1].

Corollary 2.8. *$q : T \Rightarrow T'$ is a monad morphism.*

Proof. By Lemma 2.6, we only need to show that q coincides with $\kappa \circ T\eta'$, where η' is the unit of the quotient monad. To this end consider the following diagram:

$$\begin{array}{ccccc} T & \xrightarrow{T\eta'} & TUF & \xrightarrow{\kappa} & UF \\ & \searrow^{T\eta} & \uparrow{\scriptstyle Tq} & & \uparrow{\scriptstyle q} \\ & & TT & \xrightarrow{\mu} & T \end{array}$$

Commutativity of the triangle follows from the definition of the quotient monad. For the square, notice that the components of κ are simply the quotient algebras as constructed in the proof of Lemma 2.4, and q is an algebra morphism by construction. □

Remark 2.9. As always, the monad morphism $q : T \Rightarrow T'$ induces a functor

$$\mathsf{Alg}(\mathcal{T}') \to \mathsf{Alg}(\mathcal{T}).$$

By Lemma 2.6 (2), the comparison $\mathsf{Alg}(\mathcal{T}, \mathcal{E}) \to \mathsf{Alg}(\mathcal{T}')$ followed by $\mathsf{Alg}(\mathcal{T}') \to \mathsf{Alg}(\mathcal{T})$ coincides with the inclusion $\mathsf{Alg}(\mathcal{T}, \mathcal{E}) \to \mathsf{Alg}(\mathcal{T})$.

The above construction yields a monad $\mathcal{T}'$ given a set of operations and equations. Intuitively, any monad which is isomorphic to $\mathcal{T}'$ is presented by these same operations and equations; this is captured by the following definition.

Definition 2.10. Let Σ be an endofunctor on C, $\mathcal{T}_\Sigma$ the free monad over Σ, and $\mathcal{T}'$ the quotient monad of $\mathcal{T}_\Sigma$ with respect to some $\mathcal{T}_\Sigma$-equations $\mathcal{E}$. A monad $\mathcal{K} = \langle K, \theta, \nu \rangle$ is *presented by* Σ *and* $\mathcal{E}$ if there is a monad isomorphism $i \colon T' \Rightarrow K$. ◁

Example 2.11. The *idempotent semiring monad* is defined by the functor mapping a set X to the set $\mathcal{P}_\omega(X^*)$ of finite languages over X and, for morphisms $f \colon X \to Y$ in Set we define $\mathcal{P}_\omega(f^*)(L) = \bigcup\{f(x_1) \cdots f(x_n) \mid x_1 \cdots x_n \in L\}$. Furthermore, $\eta_X \colon X \to \mathcal{P}_\omega(X^*)$ and $\mu_X \colon \mathcal{P}_\omega(\mathcal{P}_\omega(X^*)^*) \to \mathcal{P}_\omega(X^*)$ are given by

$$\begin{aligned} \eta_X(x) &= \{x\}, \\ \mu_X(\mathcal{L}) &= \textstyle\bigcup_{L_1 \cdots L_n \in \mathcal{L}} \{w_1 \cdots w_n \mid w_i \in L_i\}. \end{aligned}$$

The idempotent semiring monad is presented by two constants $\mathtt{0}$ and $\mathtt{1}$, two binary operations $+$ and $\cdot$, and the idempotent semiring axioms. The witnessing isomorphism can easily be given based on the observation that every semiring term is equivalent with respect to the idempotent semiring equations to a sum of products of variables. ◁

Finally, we discuss conditions under which $\mathsf{Alg}(\mathcal{T}, \mathcal{E})$ is isomorphic to $\mathsf{Alg}(\mathcal{T}')$. In general this need not be the case due to the fact that despite both $\mathsf{Alg}(\mathcal{T}, \mathcal{E}) \to \mathsf{Alg}(\mathcal{T})$ and $\mathsf{Alg}(\mathcal{T}) \to \mathsf{C}$ being monadic, their composition need not be monadic. This situation occurs in Example 2.3(3), where $\mathsf{Alg}(\mathcal{T}, \mathcal{E})$ is torsion-free abelian groups but $\mathsf{Alg}(\mathcal{T}') = \mathsf{Alg}(\mathcal{T})$ is abelian groups [10].

A general remark in this situation is that, due to Beck's theorem, $\mathsf{Alg}(\mathcal{T}, \mathcal{E}) \to \mathsf{C}$ is monadic if $\mathsf{Alg}(\mathcal{T}, \mathcal{E}) \to \mathsf{Alg}(\mathcal{T})$ is closed under regular epis, that is, if $A \in \mathsf{Alg}(\mathcal{T}, \mathcal{E})$ implies $B \in \mathsf{Alg}(\mathcal{T}, \mathcal{E})$ for regular epis $f : A \to B$ in $\mathsf{Alg}(\mathcal{T})$. But we can do better in a situation of special interest.

We say that C is a finitary variety if C is the category of algebras for a finitary signature and equations, or, equivalently, if C is the category of algebras for a finitary monad on Set. In particular, we have an adjoint situation $F^{\mathsf{C}} \dashv U^C : \mathsf{C} \to \mathsf{Set}$. A signature is a functor $\mathbb{N} \to \mathsf{Set}$, assigning to each 'arity' n a set of operation symbols. An endofunctor on C is

said to be generated by a signature $S : \mathbb{N} \to \mathsf{Set}$ if it is the left-Kan extension of $F^{\mathsf{C}}S$ along F^{C}, that is, if it is of the form $A \mapsto \coprod_{n\in\mathbb{N}} \mathsf{C}(F^{\mathsf{C}}n, A) \bullet F^{\mathsf{C}}Sn$. (Note that we cannot use such endofunctors to imitate Example 2.3(3) which relies on the $\mathbb{Z}_n$ not being free algebras of the form $F^{\mathsf{C}}n$.)

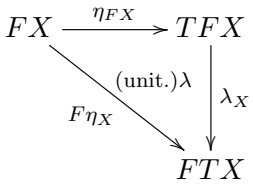
Proposition 2.12. *If in the situation described in Definition 2.10 the category* C *is a finitary variety, and* Σ *and* E *are endofunctors on* C *generated by signatures, then* $\mathsf{Alg}(\mathcal{T}, \mathcal{E})$ *is isomorphic to* $\mathsf{Alg}(\mathcal{T}')$.

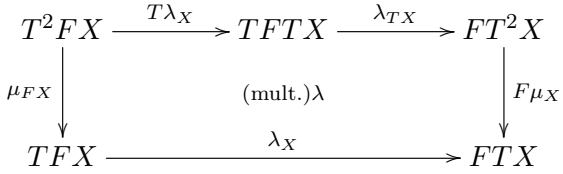
Proof. This is an instance of Theorem 4.4 of [35]. □

3. Distributive Laws and Bialgebras

We briefly recall the basic definitions of distributive laws and bialgebras; for a more thorough introduction we refer to [19, 7, 34].

3.1. **Basic Definitions.** Let $\mathcal{T} = \langle T, \eta, \mu\rangle$ be a monad on a category C, and F an endofunctor on C. A *distributive law* λ of the monad $\mathcal{T}$ over the functor F is a natural transformation $\lambda\colon TF \Rightarrow FT$ which is compatible with the monad structure, meaning that $\lambda \circ \eta_F = F\eta$ and $\lambda \circ \mu_F = F\mu \circ \lambda_T \circ T\lambda$, i.e., for all X the following diagrams commute:





We recall that every distributive law $\lambda\colon TF \Rightarrow FT$ corresponds to a *lifting* F_λ of F to the category of $\mathcal{T}$-algebras (see, e.g., [16, 19]), defined as

$$F_\lambda\langle A, \alpha\rangle = \langle FA, F\alpha \circ \lambda_A\rangle \qquad F_\lambda(f) = Ff \tag{3.1}$$

Note that the compatibility of λ_X with μ_X means precisely that λ_X is a $\mathcal{T}$-algebra homomorphism from $\langle TFX, \mu_{FX}\rangle$ to $F_\lambda\langle TX, \mu_X\rangle$.

An *F-coalgebra* is a pair $\langle X, c\rangle$ where X is a C-object and $c\colon X \to FX$ is a C-arrow. An *F-coalgebra morphism* from $\langle X, c\rangle$ to $\langle Y, d\rangle$ is an arrow $f\colon X \to Y$ such that $d \circ f = Tf \circ c$. Given a distributive law λ of $\mathcal{T}$ over F, a *λ-bialgebra* $\langle X, \alpha, \beta\rangle$ consists of a carrier X, a $\mathcal{T}$-algebra $\alpha\colon TX \to X$ and an F-coalgebra $\beta\colon X \to FX$ such that $\beta \circ \alpha = F\alpha \circ \lambda_X \circ T\beta$. A *morphism of λ-bialgebras* from $\langle X_1, \alpha_1, \beta_1\rangle$ to $\langle X_2, \alpha_2, \beta_2\rangle$ is an arrow $f\colon X_1 \to X_2$ which is both a $\mathcal{T}$-algebra homomorphism and an F-coalgebra morphism.

The following results are well known (see, e.g., [34, 7, 19]). If $\langle Z, \zeta\rangle$ is a final F-coalgebra, then a distributive law $\lambda\colon TF \Rightarrow FT$ yields a final λ-bialgebra $\langle Z, \alpha, \zeta\rangle$ where $\alpha\colon TZ \to Z$ is defined by coinduction from the F-coalgebra $\langle TZ, \lambda_Z \circ T\zeta\rangle$.

We will need the notion of distributive laws of monads over *copointed* functors. A copointed functor is a pair $\langle F, \epsilon\rangle$ where F is an endofunctor and $\epsilon\colon F \Rightarrow \mathrm{Id}$ a natural transformation. A distributive law of $\mathcal{T}$ over $\langle F, \epsilon\rangle$ is a distributive law of $\mathcal{T}$ over F additionally satisfying $\epsilon_T \circ \lambda = T\epsilon$. For any endofunctor F on a category C with products, the *cofree copointed functor generated by* F is the pair $\langle \mathrm{Id} \times F, \pi_1\colon \mathrm{Id} \times F \to \mathrm{Id}\rangle$ where π_1 is the natural left-projection.

When $\mathcal{T} = \mathcal{T}_\Sigma$ is the free monad generated by a (signature) functor Σ, then distributive laws involving $\mathcal{T}$ can be reduced to "plain" natural transformations using recursion, namely, there is a 1-1 correspondence between distributive laws $\lambda\colon T_\Sigma F \Rightarrow FT_\Sigma$ of $\mathcal{T}_\Sigma$ over F and natural transformations $\rho\colon \Sigma F \Rightarrow FT_\Sigma$ (cf. [34, 7]). Such a ρ corresponds to a specification format of operational rules, and is sometimes referred to as a *simple SOS specification*. Similarly, for cofree copointed functors, if $\mathcal{T}_\Sigma$ is freely generated by Σ, then there is a 1-1 correspondence between distributive laws $\lambda\colon T_\Sigma(\mathrm{Id} \times F) \Rightarrow (\mathrm{Id} \times F)T_\Sigma$ of $\mathcal{T}_\Sigma$ over $\langle \mathrm{Id} \times F, \pi_1 \rangle$ and natural transformations $\rho\colon \Sigma(\mathrm{Id} \times F) \Rightarrow FT_\Sigma$ (cf. [21, 13]). Such a natural transformation ρ is also referred to as an *abstract GSOS specification* since it generalises the GSOS-format for labelled transition systems where $F = (\mathcal{P}_\omega(-))^A$, cf. [7, 34]. In what follows, we will generally omit Σ-subscripts on free monads in order to keep notation uncluttered.

3.2. **Solutions to Corecursive Equations.** An important application of distributive laws is in solving *corecursive equations* which are arrows of the type $\phi\colon X \to FTX$ where F is a functor and T is (the functor component of) a monad. These include many interesting and useful structures such as linear and context-free systems of behavioural differential equations [30, 38], as well as linear, nondeterministic and weighted automata cf. [13, 32]. These are all instances of *$\mathcal{T}$-automata* [13] (where $\mathcal{T}$ is a monad on Set) which have the type $X \to B \times (TX)^A$ where A is a set and B carries a $\mathcal{T}$-algebra $\beta\colon TB \to B$, i.e., in particular, $F = B \times (-)^A$ whose final coalgebra carrier is B^{A^*}.

In the presence of a distributive law $\lambda\colon TF \Rightarrow FT$ one obtains a *λ-coinduction principle* [7] which provides unique solutions in the final λ-bialgebra $\langle Z, \alpha, \zeta \rangle$ to corecursive equations of the form $\phi\colon X \to FTX$. Ordinary coinduction is the special case where $\mathcal{T}$ is the identity monad. Formally, a solution to $\phi\colon X \to FTX$ in a λ-bialgebra $\langle A, \alpha, \beta \rangle$ is an arrow $f\colon X \to A$ such that

$$\begin{array}{ccccc} X & & \xrightarrow{\quad f \quad} & & A \\ {\scriptstyle\phi}\downarrow & & & & \downarrow{\scriptstyle\beta} \\ FTX & \xrightarrow{FTf} & FTA & \xrightarrow{F\alpha} & FA \end{array} \tag{3.2}$$

commutes. More precisely, λ-coinduction is coinduction in the category of λ-bialgebras, and we have the following fact.

Proposition 3.1 (Lemmas 4.3.3 and 4.3.4 of [7])**.** *Let $\phi\colon X \to FTX$ be a corecursive equation. Taking $\phi^\lambda = F\mu_X \circ \lambda_{TX} \circ T\phi$ then $\langle TX, \mu_X, \phi^\lambda \rangle$ is a λ-bialgebra, and $\eta_X\colon X \to TX$ is a solution of ϕ. Moreover, for any λ-bialgebra $\langle A, \alpha, \beta \rangle$, there is a 1-1 correspondence between solutions of ϕ in $\langle A, \alpha, \beta \rangle$ and λ-bialgebra morphisms from $\langle TX, \mu_X, \phi^\lambda \rangle$ to $\langle A, \alpha, \beta \rangle$.*

A "pointwise distributive law" λ for $\mathcal{T}$-automata can be obtained (cf. [13, 14]) by taking $\lambda_X = (\beta \times \mathsf{st}) \circ \langle T\pi_1, T\pi_2 \rangle$ where $\mathsf{st}\colon T \circ (-)^A \Rightarrow (-)^A \circ T$ is the strength natural transformation. This λ is called "pointwise", since the algebra structure induced on the carrier B^{A^*} of the final $B \times (-)^A$-coalgebra is the pointwise extension of $\beta\colon TB \to B$. In the context-free and streams examples below, however, the desired algebraic structure on B^{A^*} uses the convolution product which is not the pointwise extension of the semiring product of B. So for these examples a different λ must be given.

4. Quotients of Distributive Laws

In Section 2 we saw how equations give rise to quotients of algebras, and we gave a construction of the resulting quotient monad. In this section, we investigate conditions under which distributive laws and equations give rise to quotients of distributive laws.

As before, let Σ be a functor generating the free monad $\mathcal{T} = \langle T, \eta, \mu\rangle$, and let $\mathcal{E} = \langle E, l, r\rangle$ be $\mathcal{T}$-equations with the associated quotient monad $\mathcal{T}' = \langle T', \eta', \mu'\rangle$.

4.1. **Distributive Laws over Plain Behaviour Functors.** In this subsection, we assume that $\lambda\colon TF \Rightarrow FT$ is a distributive law of a monad $\mathcal{T}$ over a plain behaviour functor F. We will provide a condition on λ and the equations $\mathcal{E}$ that ensures that we get a distributive law $\lambda'\colon T'F \Rightarrow FT'$ for the quotient monad. To this end, it is convenient to use the notion of a morphism of distributive laws from [26, 36].

Definition 4.1. Let $\langle T, \eta, \mu\rangle$ and $\langle K, \theta, \nu\rangle$ be monads, and let $\lambda\colon TF \Rightarrow FT$ and $\kappa\colon KF \Rightarrow FK$ be distributive laws. A natural transformation $\tau\colon T \Rightarrow K$ is a *morphism of distributive laws* from λ to κ (notation $\tau\colon \lambda \Rightarrow \kappa$) if τ is a monad morphism and the following square commutes:

$$\begin{array}{ccc} TF & \xRightarrow{\tau F} & KF \\ {\scriptstyle\lambda}\Downarrow & & \Downarrow{\scriptstyle\kappa} \\ FT & \xRightarrow{F\tau} & FK \end{array} \tag{4.1}$$

◁

We note that there are generalisations of the above definition that allow natural transformations between behaviour functors, cf. [36]. For our purposes, we do not need to change the behaviour type.

Definition 4.2. We say that $\lambda\colon TF \Rightarrow FT$ *preserves (equations in)* $\mathcal{E}$ if for all X in C:

$$EFX \underset{r_{FX}}{\overset{l_{FX}}{\rightrightarrows}} TFX \xrightarrow{\lambda_X} FTX \xrightarrow{Fq_X} FT'X \tag{4.2}$$

commutes. ◁

In Set, preservation of equations can be conveniently formulated in terms of relation lifting. The F-lifting of a relation $R \subseteq Y \times Y$ is defined as

$$\overline{F}(R) = \{\langle F\pi_1(u), F\pi_2(u)\rangle \in FY \times FY \mid u \in F(R)\}\,.$$

For any set X, we denote by $\equiv_X$ the congruence $\ker(q_X)$ on TX generated by the equations. If the lifting $\overline{F}$ preserves inverse images, then it preserves kernel relations.[1] This means that $\overline{F}(\equiv_X) = \ker(Fq_X)$, and hence equation (4.2) is satisfied if for every set X and every $b \in EFX$:

$$\lambda_X \circ l_{FX}(b)\ \overline{F}(\equiv_X)\ \lambda_X \circ r_{FX}(b). \tag{4.3}$$

We now come to our main result. On the one hand, item (2) of Theorem 4.3 below gives us a distributive law for the quotient monad and the useful consequences that follow from it such as, e.g., the solution of recursive equations as discussed in Sections 3.2 and 5.

[1] The proof of this for polynomial functors in [15, Lemma 3.2.5(i)] goes through for arbitrary F under the assumption of preservation of inverse images, since for any F, the lifting $\overline{F}$ preserves diagonals. In particular, $\overline{F}$ preserves inverse images if F preserves weak pullbacks (see e.g., [15, Proposition 4.4.3]).

On the other hand, condition (1) in the form of (4.2) is amenable to explicit calculations as shown in Examples 4.7, 4.9, and 4.11.

Theorem 4.3. *The following are equivalent.*

(1) $\lambda\colon TF \Rightarrow FT$ *preserves equations in* $\mathcal{E}$.

(2) There is a (unique) distributive law $\lambda'\colon T'F \Rightarrow FT'$ *such that* $q\colon T \Rightarrow T'$ *is a morphism of distributive laws from* λ *to* λ'.

Proof. We first show that (4.2) extends to the following:

$$TEFX \underset{r^\sharp_{FX}}{\overset{l^\sharp_{FX}}{\rightrightarrows}} TFX \xrightarrow{\lambda_X} FTX \xrightarrow{Fq_X} FT'X \tag{4.4}$$

To obtain (4.4) it suffices to show that $Fq_X \circ \lambda_X$ is a $\mathcal{T}$-algebra homomorphism, since then $Fq_X \circ \lambda_X \circ l^\sharp_{FX}$ and $Fq_X \circ \lambda_X \circ r^\sharp_{FX}$ are $\mathcal{T}$-algebra homorphisms extending $Fq_X \circ \lambda_X \circ l_{FX}$ and $Fq_X \circ \lambda_X \circ r_{FX}$, respectively. Since these latter two are equal due to (4.2), and homomorphic extensions are unique, we then get (4.4).

We now show that $Fq_X \circ \lambda_X$ is a $\mathcal{T}$-algebra homomorphism. Let F_λ be the lifting of F to the category of $\mathcal{T}$-algebras, and recall that λ_X is a $\mathcal{T}$-algebra homomorphism from $\langle TFX, \mu_{FX}\rangle$ to $F_\lambda\langle TX, \mu_X\rangle$ (cf. Section 3.1). Since also $\langle TX, \mu_X\rangle \xrightarrow{q_X} \langle T'X, \mu'_X \circ q_{T'X}\rangle$ is a $\mathcal{T}$-algebra homomorphism, by applying the lifting F_λ we obtain a $\mathcal{T}$-algebra homomorphism

$$F_\lambda\langle TX, \mu_X\rangle \xrightarrow{Fq_X} F_\lambda\langle T'X, \mu'_X \circ q_{T'X}\rangle \ .$$

Thus $Fq_X \circ \lambda_X$ is a $\mathcal{T}$-algebra homomorphism from the free $\mathcal{T}$-algebra $\langle TFX, \mu_{FX}\rangle$.

This proves that (4.4) commutes. Now, by the universal property of the coequalizer q_{FX} there is a (unique) algebra homomorphism $\lambda'_X\colon T'FX \to FT'X$ such that $\lambda'_X \circ q_{FX} = Fq_X \circ \lambda_X$:

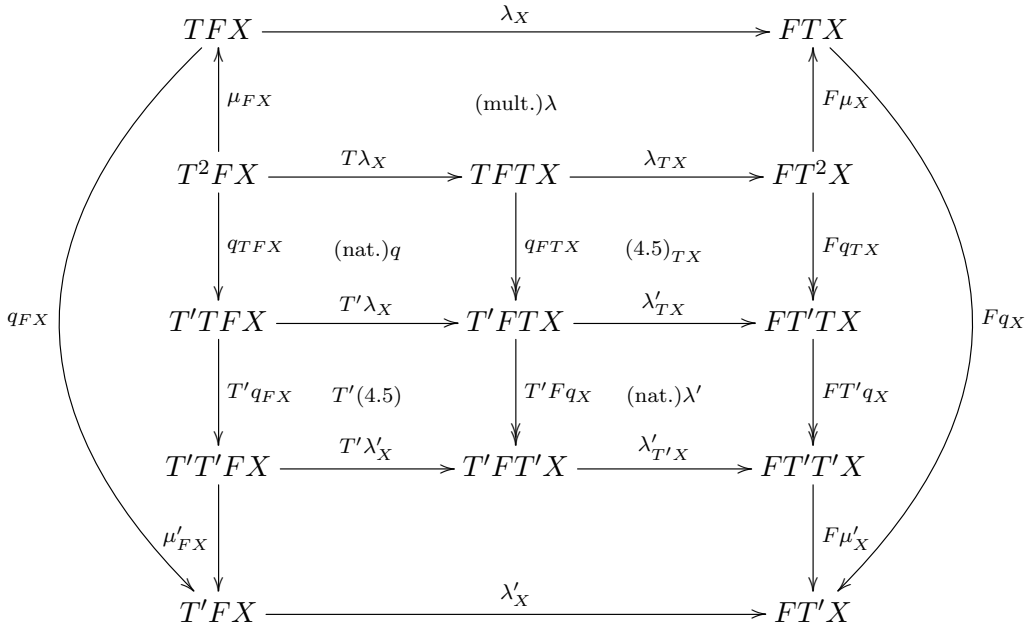
$$\begin{array}{ccccc} TEFX & \underset{r^\sharp_{FX}}{\overset{l^\sharp_{FX}}{\rightrightarrows}} & TFX & \xrightarrow{q_{FX}} & T'FX \\ & & {\scriptstyle\lambda_X}\downarrow & & \vdots\ {\scriptstyle\lambda'_X}\downarrow \\ & & FTX & \xrightarrow{Fq_X} & FT'X \end{array} \tag{4.5}$$

The naturality of λ' follows from (4.5), and the naturality of λ and q. Due to the commutativity of the square in (4.5), q is a morphism of distributive laws from λ to λ' once we show that λ' is, in fact, a distributive law.

The unit law for λ' holds due to the unit law for λ and (4.5):

$$\begin{array}{ccccc} FX & \xrightarrow{\eta_{FX}} & TFX & \xrightarrow{q_{FX}} & T'FX \\ & {\scriptstyle F\eta_X}\searrow & {\scriptstyle\lambda_X}\downarrow & (4.5) & \downarrow{\scriptstyle\lambda'_X} \\ & & FTX & \xrightarrow[Fq_X]{} & FT'X \end{array} \tag{4.6}$$

Multiplication law for λ':



(4.7)

The small upper-left square commutes by naturality of q. The small lower-left square commutes by applying T' to (4.5). The outer crescents commute since q is a monad morphism, and the outermost part does due to (4.5). Finally, use that by naturality of q, $T'q_{FX} \circ q_{TFX} = q_{T'FX} \circ Tq_{FX}$, which by Assumption 2.2 is an epi, and hence can be right-cancelled to yield commutativity of the lower rectangle as desired.

The implication from 2 to 1 follows from the fact that (4.5) implies (4.2). □

Remark 4.4. Street [33] investigates the 2-category where monads are objects and distributive laws $\lambda\colon SF \Rightarrow FT$, called monad functors, are the 1-cells $(\lambda, F)\colon T \to S$. Given a monad T, the right Kan-extension $\mathsf{Ran}_F FT$ of FT along F is again a monad. Morever, distributive laws $SF \Rightarrow FT$ are in 1-1 correspondence to monad morphisms $S \Rightarrow \mathsf{Ran}_F FT$, cf. [33, Theorem 5].

In this setting, if the free monad T_E over E exists, then the equations can be expressed more abstractly as a parallel pair of monad morphisms $T_E \rightrightarrows T$, and the distributive law $\lambda\colon TF \Rightarrow FT$ satisfies the equations iff its transpose $T \Rightarrow \mathsf{Ran}_F FT$ does, that is, iff $T_E \rightrightarrows T \longrightarrow \mathsf{Ran}_F FT \longrightarrow \mathsf{Ran}_F FT'$ commutes. Since T' is a coequaliser of monads, this induces a monad morphism $T' \Rightarrow \mathsf{Ran}_F FT'$, which, after transposing, gives a distributive law $\lambda' : T'F \Rightarrow FT'$. We thank Neil Ghani for this conceptually elegant argument of the equivalence stated in Theorem 4.3. We note that the above elementary proof does not require the existence of the free monad over E, and moreover, it avoids introducing more abstract definitions.

Remark 4.5. Using that distributive laws correspond to functor liftings on $\mathcal{T}$-algebras (cf. (3.1)), the distributive law λ' in Theorem 4.3 exists if and only if the functor F_λ restricts to $\mathcal{T}'$-algebras. A similar statement for the case when F is a monad is made in [23, Corollary 3.4.2].

As a corollary we obtain the analogue of Theorem 4.3 for monads presented by operations and equations.

Corollary 4.6. *Suppose $\mathcal{K} = \langle K, \theta, \nu\rangle$ is presented by operations Σ and equations $\mathcal{E}$ with natural isomorphism $i\colon T' \Rightarrow K$, and suppose we have a distributive law $\lambda\colon TF \Rightarrow FT$ of $\mathcal{T}$ over F. Then there exists a unique distributive law $\kappa\colon KF \Rightarrow FK$ of $\mathcal{K}$ over F such that $i \circ q\colon \lambda \Rightarrow \kappa$ is a morphism of distributive laws.*

Proof. The distributive law $\kappa\colon KF \Rightarrow KF$ is defined as $\kappa = Fi \circ \lambda \circ i^{-1}$. The proof proceeds by checking that κ indeed satisfies the defining axioms of a distributive law, which is an easy but tedious exercise. □

Theorem 4.3 says that if λ preserves the equations $\mathcal{E}$, then we can *present* λ' *as "λ modulo equations"*. We illustrate this with an example.

Example 4.7 (Stream calculus)**.** Behavioural differential equations are used extensively in [30, 31] to define streams and stream operations. Here, the behaviour functor is $FX = \mathbb{R}\times X$ whose final coalgebra $\langle \mathbb{R}^\omega, \zeta\rangle$ consists of streams over the real numbers together with the map $\zeta(\sigma) = \langle \sigma(0), \sigma'\rangle$ which maps a stream σ to its initial value $\sigma(0)$ and derivative σ'.

Consider the following system of behavioural differential equations where $[a]$, $\mathtt{X}$, σ and τ denote streams over the real numbers.

$$\begin{array}{rclrcll}
[a](0) & = & a, & [a]' & = & [0], & \forall a \in \mathbb{R} \\
\mathtt{X}(0) & = & 0, & \mathtt{X}' & = & [1], & \\
(\sigma+\tau)(0) & = & \sigma(0)+\tau(0), & (\sigma+\tau)' & = & \sigma'+\tau', & \\
(\sigma\times\tau)(0) & = & \sigma(0)\cdot\tau(0), & (\sigma\times\tau)' & = & (\sigma'\times[\tau(0)]) + ((\sigma'\times(\mathtt{X}\times\tau'))+ & \\
 & & & & & ([\sigma(0)]\times\tau')) &
\end{array} \tag{4.8}$$

The behavioural differential equations in (4.8) define the constant streams $[a] = (a, 0, 0, \ldots)$ for all $a \in \mathbb{R}$, $\mathtt{X} = (0, 1, 0, 0, \ldots)$, pointwise addition and convolution product of streams. Note that the convolution product is here defined differently than in [30, 31]. We explain this choice at the end of the example.

Since we are defining $\mathbb{R}$ many streams $[a]$, one constant stream $\mathtt{X}$ and two binary operations ($+$ and $\times$), the signature functor is $\Sigma(X) = \mathbb{R} + 1 + (X \times X) + (X \times X)$, and (4.8) corresponds to a natural transformation $\rho\colon \Sigma F \Rightarrow FT$ where T is the functor part of the free monad $\mathcal{T}$ over Σ (that is, TX is the set of all Σ-terms over variables in X). The components of ρ are given by:

$$\begin{array}{rcl}
\rho_X^{[a]} & = & \langle a, [0]\rangle \\
\rho_X^{\mathtt{X}} & = & \langle 0, [1]\rangle \\
\rho_X^{+}(\langle a, x\rangle, \langle b, y\rangle) & = & \langle a+b, x+y\rangle \\
\rho_X^{\times}(\langle a, x\rangle, \langle b, y\rangle) & = & \langle a\cdot b, (x\times[b]) + ((x\times(\mathtt{X}\times y)) + ([a]\times y))\rangle
\end{array} \tag{4.9}$$

As described at the end of section 3.1, such a ρ is a simple SOS specification, and it uniquely induces a distributive law $\lambda\colon TF \Rightarrow FT$. This λ is essentially the inductive extension of ρ from terms of depth 1 to arbitrary terms. Let $\mathcal{E}$ be given by the following axioms where $V = \{v, u, w\}$ and $a, b \in \mathbb{R}$ (see Example 2.1 for an explanation of how this corresponds to a functor with two natural transformations):

$$\begin{array}{lll}
(v+u)+w = v+(u+w) & [0]+v = v & v+u = u+v \\
(v\times u)\times w = v\times(u\times w) & [1]\times v = v & v\times u = u\times v \\
v\times(u+w) = (v\times u)+(v\times w) & [0]\times v = [0] & \\
[a+b] = [a]+[b] & [a\cdot b] = [a]\times[b] &
\end{array} \tag{4.10}$$

$\mathcal{E}$ consists of the *commutative* semiring axioms together with axioms stating the inclusion of the underlying semiring of the reals. We would like to apply Theorem 4.3 to obtain a distributive law λ' for the quotient monad $\mathcal{T}'$ arising from $\mathcal{T}$ and $\mathcal{E}$. We show that λ preserves $\mathcal{E}$. Let $\langle a, x\rangle, \langle b, y\rangle, \langle c, z\rangle \in FX$ for some set X. First note that for $F = \mathbb{R} \times \mathrm{Id}$, $\langle r_1, t_1\rangle \; \overline{F}(\equiv_X) \; \langle r_2, t_2\rangle$ iff $r_1 = r_2$ and $t_1 \equiv_X t_2$. It is straightforward to check preservation of the axioms that only concern addition, as well as of $[1] \times v = v$, $[0] \times v = [0]$ and $v \times u = u \times v$. We show that $[a \cdot b] = [a] \times [b]$ is preserved:

$$\begin{array}{rll} \lambda_X([a] \times [b]) & = & \langle a \cdot b, [0] \times [b] + [0] \times \mathtt{X} \times [0] + [a] \times [0]\rangle \\ & \overline{F}(\equiv_X) & \langle a \cdot b, [0]\rangle = \lambda_X([a \cdot b]) \end{array}$$

We check that λ preserves the distribution axiom:

$$\begin{array}{rl} & \lambda_X(\langle a, x\rangle \times (\langle b, y\rangle + \langle c, z\rangle)) \\ = & \langle a \cdot (b + c), (x \times [b + c]) + (x \times X \times (y + z)) + [a] \times (y + z)\rangle \\ \overline{F}(\equiv_X) & \langle a \cdot (b + c), (x \times [b + c]) + (x \times X \times y) + (x \times X \times z) + \\ & \qquad ([a] \times y) + ([a] \times z)\rangle \\ \overline{F}(\equiv_X) & \langle (a \cdot c) + (b \cdot c), (x \times [b]) + (x \times X \times y) + ([a] \times y) + \\ & \qquad (x \times [c]) + (x \times X \times z) + ([a] \times z)\rangle \\ = & \\ \lambda_X((\langle a, x\rangle \times \langle b, y\rangle) + (\langle a, x\rangle \times \langle c, z\rangle)) & \end{array}$$

Note that we used $[a + b] = [a] + [b]$. Similarly, preservation of $\times$-associativity can be verified, and it uses the axiom $[a \cdot b] = [a] \times [b]$. We have thus shown that λ preserves $\mathcal{E}$, and it follows, in particular, that $\langle \mathbb{R}^\omega, +, \times, [0], [1]\rangle$ is a commutative semiring. This was shown directly in [31], but the proof uses bisimulation-up-to as well as the fundamental theorem of stream calculus, which cannot be added as an equation. In our approach we construct a distributive law, and obtain not only this result but also the soundness of the bisimulation-up-to technique [29], and the existence of unique solutions to corecursive equations $\phi\colon X \to FT'X$ (see Section 3.2).

The derivative of the convolution product is usually (cf. [30, 31]) specified as:

$$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau') \tag{4.11}$$

which corresponds to a stream GSOS-rule $\Sigma(\mathrm{Id} \times \mathbb{R} \times \mathrm{Id}) \Rightarrow \mathbb{R} \times T(-)$, and thus to a distributive law over the cofree copointed functor. However, with this definition, we could not show that the commutativity of $\times$ is preserved although all other axioms remain preserved. Hence a given λ does not necessarily satisfy all equations that are valid on the final F-coalgebra. ◁

In the above example the monad under consideration is defined by operations and equations. In Example 4.11 below we will see an example of a monad that has an independent definition, but where a presentation by operations and equations simplifies the construction of a distributive law considerably.

Remark 4.8. The concrete proof method for preservation of equations bears a close resemblance to *bisimulation up to congruence* [29], in that one must show that for every pair in the (image of the) equations, its derivatives are related by the least *congruence* $\equiv_X$ instead of just the equivalence relation induced by the equations.

Example 4.9. As we discussed at the end of Example 4.7 (regarding the definition of the convolution product), it is not always possible to show that a given λ preserves all equations that hold in the final coalgebra. Now we give another concrete example of this fact. This

example again concerns stream systems, i.e., coalgebras for the functor $FX = \mathbb{R} \times X$. We define the constant stream of zeros by three different constants n_1, n_2 and n_3 by the following behavioural differential equations:

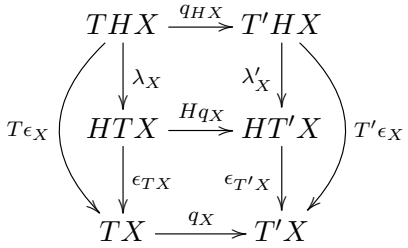
$$n_1(0) = 0, \quad n_1' = n_1 \qquad n_2(0) = 0, \quad n_2' = n_3 \qquad n_3(0) = 0, \quad n_3' = n_3$$

The corresponding signature functor is thus $\Sigma X = 1 + 1 + 1$, and the above specification gives rise to a distributive law $\lambda\colon TF \Rightarrow FT$ where T is (the functorial component of) the free monad over Σ. Now consider the equation $n_1 = n_2$; this clearly holds when interpreted in the final coalgebra. However, this equation is not preserved by λ. To see this, notice that $\lambda(n_1) = \langle 0, n_1\rangle$ and $\lambda(n_2) = \langle 0, n_3\rangle$, but $n_1 \not\equiv_X n_3$, so $\lambda(n_1)$ and $\lambda(n_2)$ are not related by $\overline{F}(\equiv_X)$. ◁

4.2. **Distributive Laws over Copointed Functors.** We now show that our main results hold as well for distributive laws of monads over *copointed* functors. This extends our method to deal with operations specified in the abstract GSOS format, such as language concatenation.

Proposition 4.10. *Theorem 4.3 and Corollary 4.6 hold as well for any distributive law of a monad over a copointed functor.*

Proof. Let $\langle H, \epsilon\rangle$ be a copointed functor and $\lambda\colon TH \Rightarrow HT$ a distributive law of $\mathcal{T}$ over $\langle H, \epsilon\rangle$. Suppose λ preserves equations E. By Theorem 4.3 then there is a distributive law λ' of $\mathcal{T}'$ over H such that $q\colon T \Rightarrow T'$ is a morphism of distributive laws. In order to show that λ' is a distributive law of $\mathcal{T}'$ over $\langle H, \epsilon\rangle$ we only need to prove that λ' satisfies the additional axiom, i.e., that the right crescent in the following diagram commutes:



The outermost part commutes by naturality of q, the two squares commute by naturality of λ and ϵ, and the left crescent commutes by the fact that λ is a distributive law of $\mathcal{T}$ over $\langle H, \epsilon\rangle$. Consequently we have $\epsilon_{T'X} \circ \lambda'_X \circ q_{HX} = T'\epsilon_X \circ q_{HX}$, and since q_{HX} is an epi we obtain $\epsilon_{T'X} \circ \lambda'_X = T'\epsilon_X$ as desired.

For Corollary 4.6 one needs to add to its proof a check that the distributive law satisfies the additional axiom as well, which is again rather easy to do. □

Example 4.11 (Context-free languages)**.** A context free grammar (in Greibach normal form) consists of a finite set A of terminal symbols, a (finite) set X of non-terminal symbols, and a map $\langle o, t\rangle\colon X \to 2 \times \mathcal{P}_\omega(X^*)^A$, i.e., it is a coalgebra for the behavior functor $F(X) = 2 \times X^A$ composed with the idempotent semiring monad $\mathcal{P}_\omega((-)^*)$ from Example 2.11. Intuitively, $o(x) = 1$ means that the variable x can generate the empty word, whereas $w \in t(x)(a)$ if and only if x can generate aw, cf. [38].

It is a rather difficult task to describe concretely a distributive law of $T' = \mathcal{P}_\omega((-)^*)$ over F (or $\mathrm{Id} \times F$) defining the sum $+$ and sequential composition $\cdot$ of context-free grammars.

More conveniently, since we have seen in Example 2.11 that the monad $\mathcal{P}_\omega((-)^*)$ can be presented by the operations and axioms of idempotent semirings, we proceed by defining a distributive law λ of the free monad $\mathcal{T}_\Sigma$ generated by the semiring signature functor $\Sigma(X) = 1+1+(X\times X)+(X\times X)$ over the cofree copointed functor $\langle \mathrm{Id}\times F, \pi_1\rangle$, and show that λ preserves the semiring axioms. We define λ as the distributive law that corresponds to the natural transformation $\rho\colon \Sigma(\mathrm{Id}\times F)\Rightarrow FT$ whose components are given by:

$$\begin{array}{rcl} \rho^{\mathsf{0}}_X & = & \langle 0, a\mapsto \emptyset\rangle \\ \rho^{\mathsf{1}}_X & = & \langle 1, a\mapsto \emptyset\rangle \\ \rho^{+}_X(\langle x,o,f\rangle,\langle y,p,g\rangle) & = & \langle \max\{o,p\}, a\mapsto f(a)+g(a)\rangle \\ \rho^{\cdot}_X(\langle x,o,f\rangle,\langle y,p,g\rangle) & = & \left\langle \min\{o,p\}, a\mapsto \begin{cases} f(a)\cdot y & \text{if } p=0 \\ f(a)\cdot y + g(a) & \text{if } p=1 \end{cases}\right\rangle \end{array} \tag{4.12}$$

We proceed to show that λ preserves the defining equations of idempotent semirings. We treat here only the case of distributivity, i.e., $u\cdot(v+w) = u\cdot v + u\cdot w$. To this end, let X be arbitrary and suppose $\langle x,o,d\rangle, \langle y,p,e\rangle, \langle z,q,f\rangle \in X\times FX$. Notice that either $o=0$ or $o=1$; we treat both cases separately:

$$\begin{array}{rl} & \lambda(\langle x,0,d\rangle\cdot(\langle y,p,e\rangle+\langle z,q,f\rangle)) \\ = & (x\cdot(y+z), 0, a\mapsto d(a)\cdot(y+z)) \\ \overline{F}(\equiv_X) & (x\cdot y + x\cdot z, 0, a\mapsto d(a)\cdot y + d(a)\cdot z) \\ = & \lambda(\langle x,0,d\rangle\cdot\langle y,p,e\rangle + \langle x,0,d\rangle\cdot\langle z,q,f\rangle) \end{array}$$

$$\begin{array}{rl} & \lambda(\langle x,1,d\rangle\cdot(\langle y,p,e\rangle+\langle z,q,f\rangle)) \\ = & (x\cdot(y+z), p+q, a\mapsto d(a)\cdot(y+z) + (e(a)+f(a))) \\ \overline{F}(\equiv_X) & (x\cdot y + x\cdot z, p+q, a\mapsto (d(a)\cdot y + d(a)\cdot z) + (e(a)+f(a))) \\ \overline{F}(\equiv_X) & (x\cdot y + x\cdot z, p+q, a\mapsto (d(a)\cdot y + e(a)) + (d(a)\cdot z + f(a))) \\ = & \lambda(\langle x,1,d\rangle\cdot\langle y,p,e\rangle + \langle x,1,d\rangle\cdot\langle z,q,f\rangle)\,. \end{array}$$

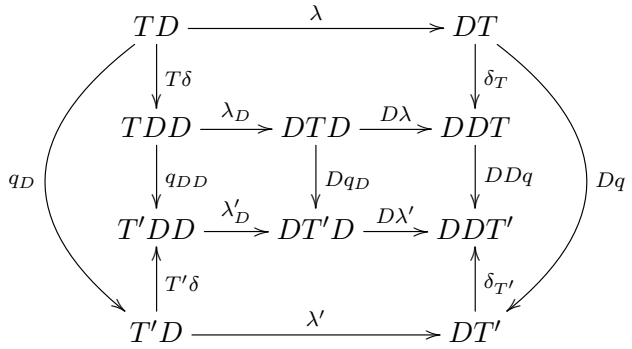
In a similar way one can show that λ preserves the other idempotent semiring equations. Thus, from Proposition 4.10 and Corollary 4.6 we obtain a distributive law κ of $\mathcal{P}_\omega((-)^*)$ over $\mathrm{Id}\times F$ such that $i\circ q\colon \lambda\Rightarrow\kappa$ is a morphism of distributive laws, i.e., κ is presented by λ and the equations of idempotent semirings. ◁

4.3. **Distributive Laws over Comonads.** A further type of distributive law, which generalizes all of the above, is that of a distributive law of a monad over a comonad. These arise from GSOS laws as well as from *coGSOS* laws, which allow to model operational rules which involve look-ahead in the premises. We refer to [19] for technical details and an example of a coGSOS format on streams. In this subsection, we prove for future reference that when constructing the quotient distributive law as above for a distributive law over a comonad, the axioms are preserved, i.e., the quotient is again a distributive law over the comonad.

Proposition 4.12. *Theorem 4.3 and Corollary 4.6 hold as well for any distributive law of a monad over a comonad.*

Proof. Let $\langle D,\epsilon,\delta\rangle$ be a comonad and $\lambda\colon TD\Rightarrow DT$ a distributive law of the monad $\langle T,\eta,\mu\rangle$ over the comonad $\langle D,\epsilon,\delta\rangle$. Suppose λ preserves equations $\mathcal{E}$. By Proposition 4.10 there is a

distributive law λ' of $\mathcal{T}'$ over the copointed functor $\langle D, \epsilon\rangle$. To show that λ' is a distributive law over the comonad $\langle D, \epsilon, \delta\rangle$, we need to check that the corresponding axiom holds.



The outermost part and the right square both commute by the fact that q is a morphism of distributive laws. The outer crescents commute since q and δ are natural. The small rectangles commute since q is a morphism from λ to λ'. The upper rectangle commutes by the assumption that λ is a distributive law over the comonad. Checking that the lower rectangle commutes, which is what we need to prove, is now an easy diagram chase, using that q_D is epic. □

5. Morphisms and Solutions

In this section, we show that morphisms of distributive laws commute with solving corecursive equations. In the case of monads with equations, this means that first solving equations ϕ with respect to $\mathcal{T}$ and then forming the quotient of the solution bialgebra is the same as first forming the quotient of $\mathcal{T}$ and solving with respect to the quotient monad $\mathcal{T}'$.

We first describe some functors that link the relevant categories of bialgebras and corecursive equations. Throughout this Section, we let $\mathcal{T} = \langle T, \eta, \mu\rangle$ and $\mathcal{K} = \langle K, \theta, \nu\rangle$ be monads; and $\lambda\colon TF \Rightarrow FT$ and $\kappa\colon KF \Rightarrow FK$ be distributive laws of $\mathcal{T}$ and $\mathcal{K}$ over F, respectively.

If $\tau\colon \lambda \Rightarrow \kappa$ is a morphism of distributive laws, then precomposing with τ yields a functor:

$$\begin{array}{rccc} I\colon & \mathsf{Bialg}(\kappa) & \to & \mathsf{Bialg}(\lambda) \\ & KX \xrightarrow{\alpha} X \xrightarrow{\beta} FX & \mapsto & TX \xrightarrow{\alpha\circ\tau_X} X \xrightarrow{\beta} FX \end{array} \tag{5.1}$$

It follows from the naturality of τ and $F\tau \circ \lambda = \kappa \circ \tau F$ that I takes a κ-bialgebra to a λ-bialgebra. Similarly, postcomposing with $F\tau$ yields a functor between corecursive equations:

$$\begin{array}{rccc} Q\colon & \mathsf{Coalg}(FT) & \to & \mathsf{Coalg}(FK) \\ & \phi\colon X \to FTX & \mapsto & F\tau_X \circ \phi\colon X \to FKX \end{array} \tag{5.2}$$

Recall from Section 3.2, that given a distributive law $\lambda\colon TF \Rightarrow FT$, the solutions of a corecursive equation $\phi\colon X \to FTX$ are characterised by morphisms from the λ-bialgebra $\langle TX, \mu_X, \phi^\lambda\rangle$ whose F-coalgebra structure given by

$$\phi^\lambda = F\mu_X \circ \lambda_{TX} \circ T\phi \tag{5.3}$$

This yields a functor (see, e.g.,[15, Lem. 5.4.11]):

$$\begin{array}{rccc} G_\lambda\colon & \mathsf{Coalg}(FT) & \to & \mathsf{Bialg}(\lambda) \\ & \langle X, \phi\rangle & \mapsto & \langle TX, \mu_X, \phi^\lambda\rangle \end{array} \tag{5.4}$$

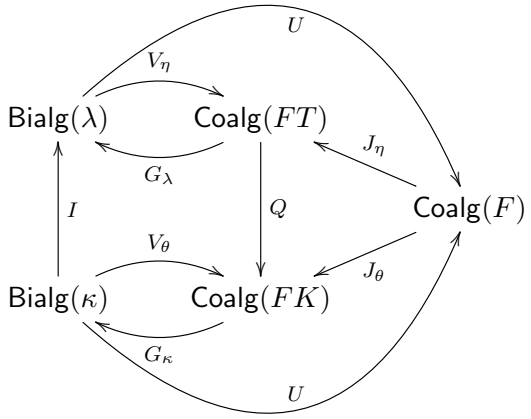
We can go in the opposite direction by using the monad unit,

$$\begin{array}{rccc} V_\eta\colon & \mathsf{Bialg}(\lambda) & \to & \mathsf{Coalg}(FT) \\ & \langle X, \alpha, \beta\rangle & \mapsto & \langle X, F\eta_X \circ \beta\rangle \end{array} \tag{5.5}$$

which decomposes into the functor $U\colon \mathsf{Bialg}(\lambda) \to \mathsf{Coalg}(F)$ that forgets algebra structure, and

$$\begin{array}{rccc} J_\eta\colon & \mathsf{Coalg}(F) & \to & \mathsf{Coalg}(FT) \\ & \langle X, \beta\rangle & \mapsto & \langle X, F\eta_X \circ \beta\rangle \end{array} \tag{5.6}$$

The following diagram summarises the situation:



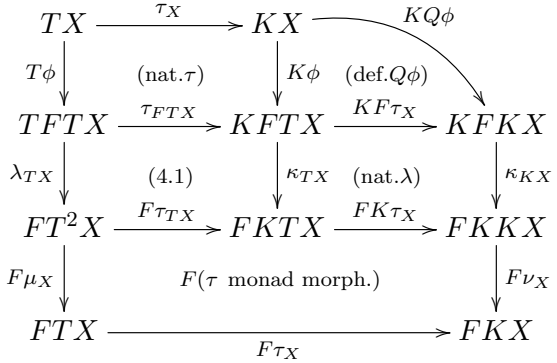
(5.7)

We mention that $QV_\eta I = V_\theta$ since τ is compatible with the units of $\mathcal{T}$ and $\mathcal{K}$.

Morphisms of distributive laws are defined to be monad maps, and hence respect the algebraic structure. The next proposition shows that, as one might expect, they also respect the coalgebraic structure, and hence morphisms of distributive laws induce morphisms between bialgebras.

Proposition 5.1. *If $\tau\colon \lambda \Rightarrow \kappa$ is a morphism of distributive laws, then for all $\phi : X \to FTX$ we have that τ_X is a λ-bialgebra morphism $\tau_X\colon G_\lambda(\phi) \to IG_\kappa Q(\phi)$ or, equivalently, an F-coalgebra morphism $\tau_X\colon \langle TX, \phi^\lambda\rangle \to \langle KX, (Q\phi)^\kappa\rangle$.*

Proof. We show that $\tau_X\colon \langle TX, \phi^\lambda\rangle \to \langle KX, (Q\phi)^\kappa\rangle$ is an F-coalgebra morphism:



□

It follows that the unique λ-bialgebra morphism $g\colon \langle TX, \mu_X, \phi^\lambda\rangle \to \langle Z, \alpha, \zeta\rangle$ into the final λ-bialgebra $\langle Z, \alpha, \zeta\rangle$ factors as $g = g' \circ \tau_X$, where g' is the final λ-bialgebra morphism from $IG_\kappa Q(\phi)$, as shown here:

$$\begin{array}{ccccccc} & & T^2X & \xrightarrow{T\tau_X} & TKX & \xrightarrow{Tg'} & TZ \\ & & \downarrow{\scriptstyle \mu_X} & & \downarrow{\scriptstyle \nu_X\circ\tau_{KX}} & & \downarrow{\scriptstyle \alpha} \\ X & \xrightarrow{\eta_X} & TX & \xrightarrow{\tau_X} & KX & \xrightarrow{g'} & Z \\ & \searrow^{\phi} & \downarrow{\scriptstyle \phi^\lambda} & & \downarrow{\scriptstyle (Q\phi)^\kappa} & & \downarrow{\scriptstyle \zeta} \\ & & FTX & \xrightarrow{F\tau_X} & FKX & \xrightarrow{Fg'} & FZ \end{array} \tag{5.8}$$

Hence by Proposition 3.1, every solution of ϕ in the final λ-bialgebra yields a solution of $Q\phi$, and vice versa.

When $\tau\colon \lambda \Rightarrow \kappa$ arises from a set of preserved equations $\mathcal{E}$ as in Section 4 (with $\kappa = \lambda'$), then Proposition 5.1 says that $IG_\kappa Q(\phi)$ is a quotient of the "free" λ-bialgebra $\langle TX, \mu_X, \phi^\lambda\rangle$, and in particular, the congruence $\equiv_X$ is an F-behavioural equivalence. In this case, $Q\phi$ is the corecursive equation obtained by reading the right-hand side of ϕ modulo equations in E. In other words, forming the quotient of the solution of the equation ϕ is the same as solving the quotiented equation $Q\phi$.

Example 5.2. Recall from Example 4.11 that $i \circ q\colon T \Rightarrow \mathcal{P}_\omega(X^*)$ is a morphism of distributive laws. By Proposition 5.1 we have the following commuting diagram for any corecursive equation $\phi\colon X \to 2 \times (TX)^A$:

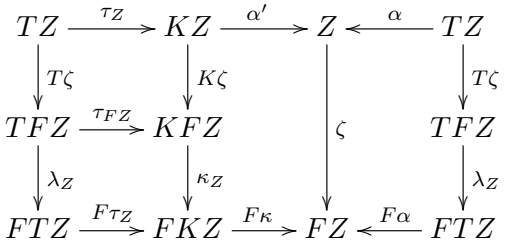
$$\begin{array}{ccccccc} X & \xrightarrow{\eta_X} & TX & \xrightarrow{(i\circ q)_X} & \mathcal{P}_\omega(X^*) & \longrightarrow & \mathcal{P}(A^*) \\ & \searrow^{\phi} & \downarrow{\scriptstyle \phi^\lambda} & & \downarrow{\scriptstyle (Q\phi)^\kappa} & & \downarrow{\scriptstyle \zeta} \\ & & 2\times(TX)^A & \xrightarrow{\mathrm{id}\times((i\circ q)_X)^A} & 2\times(\mathcal{P}_\omega(X^*))^A & \longrightarrow & 2\times\mathcal{P}(A^*)^A \end{array} \tag{5.9}$$

Notice that a context-free grammar $\langle o, t\rangle\colon X \to 2 \times \mathcal{P}_\omega(X^*)^A$ can be represented by a $\phi\colon X \to 2 \times (TX)^A$ such that $Q\phi = \langle o, t\rangle$, since $i \circ q$ is surjective. This gives the expected correspondence between two of the three different coalgebraic approaches to context-free languages introduced in [38] (the third approach is about fixed-point expressions and as such is outside the scope of this paper). ◁

Similarly, the algebraic structure induced by λ on the final F-coalgebra factors uniquely through the algebraic structure induced by κ.

Proposition 5.3. *Let $\tau\colon \lambda \Rightarrow \kappa$ be a morphism of distributive laws, and let $\alpha\colon TZ \to Z$ and $\alpha'\colon KZ \to Z$ be the algebras induced by λ and κ respectively on the final coalgebra $\langle Z, \zeta\rangle$. Then $\alpha = \alpha' \circ \tau_Z$.*

Proof. Consider the following diagram:



The upper left square commutes by naturality of τ, whereas the lower left square commutes since τ is a morphism of distributive laws. The two rectangles commute by definition of α and α' (see Section 3). Thus $\alpha' \circ \tau_Z$ and α are both coalgebra homomorphisms from $\langle TZ, \lambda_Z \circ T\zeta \rangle$ to $\langle Z, \zeta \rangle$ and consequently $\alpha' \circ \tau_Z = \alpha$ by finality. □

Example 5.4. Continuing Example 5.2, it follows from Proposition 5.3 that the algebra $\alpha \colon T\mathcal{P}(A^*) \to \mathcal{P}(A^*)$ induced by the distributive law for $\mathcal{T}$ can be decomposed as $i \circ q \circ \alpha'$, where α' is the algebra on $\mathcal{P}(A^*)$ induced by the distributive law for $\mathcal{P}_\omega(\mathrm{Id}^*)$. It can be shown by induction that α is the algebra on languages given by union and concatenation product. Now $\alpha' \colon \mathcal{P}_\omega(\mathcal{P}(A^*)^*) \to \mathcal{P}(A^*)$ can be given by selecting a representative term and applying α, and it follows that $\alpha'(\mathcal{L}) = \bigcup_{L_1 \cdots L_n \in \mathcal{L}} \{w_1 \cdots w_n \mid w_i \in L_i\}$. ◁

6. Discussion and Conclusion

We have presented a preservation condition that is necessary and sufficient for the existence of a distributive law λ' for a monad with equations given a distributive law λ for the underlying free monad. This condition consists of checking that the base equations are preserved by λ. Example 4.11 shows that presenting a monad by operations and equations and then checking that λ preserves the equations can be much easier than describing and verifying the distributive law requirements directly. We demonstrated our method by applying it to obtain distributive laws for stream calculus over commutative semirings, and for context-free grammars which use the monad of idempotent semirings.

In [36] the notion of morphisms of distributive laws is studied as a general approach to translations between operational semantics. In this paper we investigate in detail the case of quotients of distributive laws. Distributive laws for monad quotients and equations are also studied in [21, 23]. The setting and motivation of [23] is different as they study distributive laws of one monad over another with the aim to compose these monads. We study distributive laws of a monad over a plain functor, a copointed functor or a comonad. The approach in [21] differs from ours in that the desired distributive law is contingent on two given distributive laws and the existence of the coequaliser (in the category of monads) which encodes equations. We have given a more direct analysis for monads in Set and a practical proof principle, which covers many known examples. We leave as future work to find out precisely how their Theorem 31 relates to our Theorem 4.3. In [1] effects with equations are added to the syntax generated by a free monad T, using as semantic domain a suitable final B-coalgebra in the Kleisli category of T (assumed to be enriched over ω-complete pointed partial-orders). To prove adequacy of the semantics with respect to a given operational model, the authors use a result similar to our Theorem 4.3. Their result, however, is limited to coalgebras for the functor $BX = V + X$. Moreover, since we work

in Eilenberg-Moore categories of algebras rather than Kleisli categories of free algebras, we do not need to require the monad (and the quotient map) to be strong.

While in this work we have focused on adding equations which already hold in the final bialgebra, it is often useful to use equations to *induce* behaviour, next to a behavioural specification in terms of a distributive law. In process theory this idea is captured by the notion of structural congruences [25]. At the more general level of distributive laws there is work on adding recursive equations [18]. A study of structural congruences for distributive laws on free monads was given recently in [28]. While that work focuses only on free monads, we believe that it can possibly be combined with the present work to give a more general account of equations and structural congruences for different monads.

In the case of GSOS on labelled transition systems, proving equations to hold at the level of a specification was considered in [2], based on *rule-matching bisimulations*, a refinement of De Simone's notion of FH-bisimulation. Rule-matching bisimulations are based on the syntactic notion of *ruloids*, while our technique is based on preservation of equations at the level of distributive laws. It is currently not clear what the precise relation between these two approaches is; one difference is that preserving equations naturally incorporates reasoning up to congruence.

More technically, it remains an open problem whether a converse of Proposition 5.1 holds. We intend to investigate this matter in future work.

References

[1] F. Abou-Saleh and D. Pattinson. Comodels and effects in mathematical operational semantics. In F. Pfenning, editor, *Proceedings of FOSSACS 2013*, volume 7794 of *Lecture Notes in Computer Science*, pages 129–144. Springer, 2013.

[2] L. Aceto, M. Cimini, and A. Ingólfsdóttir. Proving the validity of equations in GSOS languages using rule-matching bisimilarity. *Mathematical Structures in Computer Science*, 22(2):291–331, 2012.

[3] L. Aceto, W.J. Fokkink, and C. Verhoef. Structural operational semantics. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, pages 197–292. Elsevier, 2001.

[4] J. Adámek, J. Rosický, and E. Vitale. *Algebraic Theories*. Cambridge University Press, 2011.

[5] Jiří Adámek, Václav Koubek, and Jiří Velebil. A duality between infinitary varieties and algebraic theories. *Commentationes Mathematicae Universitatis Carolinae*, 41(3):529–542, 2000.

[6] M. Barr and C. Wells. Toposes, theories, and triples. Reprints in Theory and Applications of Categories, No. 12, 2005. Available at `http://www.tac.mta.ca/tac/reprints/articles/12/tr12abs.html`.

[7] F. Bartels. *On Generalised Coinduction and Probabilistic Specification Formats*. PhD thesis, Vrije Universiteit Amsterdam, 2004.

[8] S.L. Bloom and J.B. Wright. P-varieties - a signature independent characterization of varieties of ordered algebras. *Journal of Pure and Applied Algebra*, 29(1):13 – 58, 1983.

[9] M.M. Bonsangue, H.H. Hansen, A. Kurz, and J. Rot. Presenting distributive laws. In R. Heckel and S. Milius, editors, *Proceedings of CALCO 2013*, volume 8089 of *Lecture Notes in Computer Science*, pages 95–109. Springer, 2013.

[10] F. Borceux. *Handbook of Categorical Algebra 2: Categories and Structure*. Cambridge University Press, 1994.

[11] E. Dubuc. *Kan Extensions in Enriched Category Theory*, volume 145 of *Lecture Notes in Mathematics*. Springer-Verlag, 1970.

[12] H.H. Hansen and B. Klin. Pointwise extensions of GSOS-defined operations. *Mathematical Structures in Computer Science*, 21(2):321–361, 2011.

[13] B. Jacobs. A bialgebraic review of deterministic automata, regular expressions and languages. In K. Futatsugi, J.-P. Jouannaud, and J. Meseguer, editors, *Algebra, Meaning and Computation*, volume 4060 of *Lecture Notes in Computer Science*, pages 375–404. Springer, 2006.

[14] B. Jacobs. Distributive laws for the coinductive solution of recursive equations. *Inf. Comput.*, 204(4):561–587, 2006.

[15] B. Jacobs. Introduction to coalgebra: Towards mathematics of states and observations. version 2.0., 2012. Unpublished book draft.
[16] P.T. Johnstone. Adjoint lifting theorems for categories of algebras. *Bull. London Math. Society*, 7:294–297, 1975.
[17] G.M. Kelly. A unified treatment of transfinite constructions for free algebras, free monoids, colimits, associated sheaves, and so on. *Bull. Austral. Math. Soc.*, 22:1–84, 1980.
[18] B. Klin. Adding recursive constructs to bialgebraic semantics. *J. Logic and Algebraic Programming*, 60-61:259–286, 2004.
[19] B. Klin. Bialgebras for structural operational semantics: An introduction. *Theoretical Computer Science*, 412:5043–5069, 2011.
[20] F.W. Lawvere. *Functorial Semantics of Algebraic Theories*. PhD thesis, Columbia University, 1963. Available as *Reprints in Theory and Applications of Categories*, No. 5.
[21] M. Lenisa, J. Power, and H. Watanabe. Category theory for operational semantics. *Theoretical Computer Science*, 327(1-2):135–154, 2004.
[22] F.E.J. Linton. An outline of functorial semantics. In B. Eckmann and M. Tierny, editors, *Seminar on Triples and Categorical Homology Theory*, volume 80 of *Lecture Notes in Mathematics*. Springer-Verlag, 1969. Available as *Reprints in Theory and Applications of Categories*, No. 18.
[23] E. Manes and P. Mulry. Monad compositions I: General constructions and recursive distributive laws. *Theory and Applications of Categories*, 18(7):172–208, 2007.
[24] S. Milius. A sound and complete calculus for finite stream circuits. In *Proceedings of LICS 2012*, pages 421–430. IEEE Computer Society, 2010.
[25] M.R. Mousavi and M.A. Reniers. Congruence for structural congruences. In V. Sassone, editor, *Proceedings of FoSSaCS 2005*, volume 3441 of *Lecture Notes in Computer Science*, pages 47–62. Springer, 2005.
[26] J. Power and H. Watanabe. Combining a monad and a comonad. *Theoretical Computer Science*, 280:137–162, 2002.
[27] J. Rot, F. Bonchi, M. Bonsangue, D. Pous, J. Rutten, and A. Silva. Enhanced coalgebraic bisimulation. *To appear in MSCS*, 2014. Available at `http://www.liacs.nl/~jrot/up-to.pdf`.
[28] J. Rot and M. M. Bonsangue. Combining bialgebraic semantics and equations. In A. Muscholl, editor, *Proceedings of FOSSACS 2014*, volume 8412 of *Lecture Notes in Computer Science*, pages 381–395. Springer, 2014.
[29] J. Rot, M.M. Bonsangue, and J.J.M.M. Rutten. Coalgebraic bisimulation-up-to. In P. van Emde Boas, F.C.A. Groen, G.F. Italiano, J.R. Nawrocki, and H. Sack, editors, *Proceedings of SOFSEM 2013*, volume 7741 of *Lecture Notes in Computer Science*, pages 369–381. Springer, 2013.
[30] J.J.M.M. Rutten. Behavioural differential equations: a coinductive calculus of streams, automata and power series. *Theoretical Computer Science*, 308(1):1–53, 2003.
[31] J.J.M.M. Rutten. A coinductive calculus of streams. *Mathematical Structures in Computer Science*, 15:93–147, 2005.
[32] A. Silva, F. Bonchi, M.M. Bonsangue, and J.J.M.M. Rutten. Generalizing the powerset construction, coalgebraically. In K. Lodaya and M. Mahajan, editors, *Proceedings of FSTTCS 2010*, volume 8 of *LIPIcs*, pages 272–283. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010.
[33] R. Street. The formal theory of monads. *Journal of Pure and Applied Algebra*, 2(2):149–168, 1972.
[34] D. Turi and G.D. Plotkin. Towards a mathemathical operational semantics. In *Proceedings of LICS'97*, pages 280–291. IEEE Computer Society, 1997.
[35] J. Velebil and A. Kurz. Equational presentations of functors and monads. *Mathematical Structures in Computer Science*, 21(2):363–381, 2011.
[36] H. Watanabe. Well-behaved translations between structural operational semantics. In L. Moss, editor, *Proceedings of CMCS 2002*, volume 65 of *ENTCS*, pages 337–357. Elsevier, 2002.
[37] G. Winskel. *The formal semantics of programming languages - an introduction*. Foundation of computing series. MIT Press, 1993.
[38] J. Winter, M. Bonsangue, and J. Rutten. Context-free languages, coalgebraically. In A. Corradini, B. Klin, and C. Cirstea, editors, *Proceedings of CALCO 2011*, volume 6859 of *Lecture Notes in Computer Science*, pages 359–376. Springer, 2011.