Structural Congruence for Bialgebraic Semantics[☆]

Jurriaan Rot

Radboud University, Toernooiveld 212, Nijmegen, The Netherlands

Marcello Bonsangue

LIACS – Leiden University, Niels Bohrweg 1, Leiden, The Netherlands

Centrum Wiskunde en Informatica, Science Park 123, Amsterdam, The Netherlands

Abstract

It was observed by Turi and Plotkin that structural operational semantics can be studied at the level of universal coalgebra, providing specification formats for well-behaved operations on many different types of systems. We extend this framework with non-structural assignment rules which can express, for example, the syntactic format for structural congruences proposed by Mousavi and Reniers. Our main result is that the operational model of such an extended specification is well-behaved, in the sense that bisimilarity is a congruence and that bisimulation-up-to techniques are sound.

Keywords: structural congruence, coalgebra, structural operational semantics

1. Introduction

Structural operational semantics (SOS) is a framework for defining the semantics of programming languages and process calculi in terms of transition system specifications [1]. By imposing syntactic restrictions on the type of specifications, one can prove well-behavedness properties of transition systems at the meta-level of their specification. For instance, any specification in the GSOS format [4] has a unique operational model, on which bisimilarity is a congruence.

Traditionally, research in SOS has focused on labelled transition systems as the fundamental model of behaviour. Turi and Plotkin [42] introduced the *bialgebraic* approach to structural operational semantics, where in particular GSOS can be studied at the level of *universal coalgebra* [37]. The theory of

[☆]This article is a revised and extended version of a FoSSaCS 2014 paper [35]. An extended abstract was presented at the 25th Nordic Workshop on Programming Theory, NWPT 2013, in Tallinn.

Email addresses: jrot@cs.ru.nl (Jurriaan Rot), m.m.bonsangue@liacs.leidenuniv.nl (Marcello Bonsangue)

coalgebras provides a mathematical framework for the uniform study of many types of state-based systems, including labelled transition systems but also, e.g., (non)-deterministic automata, stream systems and various types of probabilistic and weighted automata [38, 22, 3]. In the coalgebraic framework, there is a canonical notion of bisimilarity, which instantiates to the classical definition of (strong) bisimilarity in the case of labelled transition systems. It is shown in [42] that GSOS specifications can be generalised by certain natural transformations, which are called *abstract GSOS specifications*, and that these correspond to the categorical notion of *distributive laws*. This provides enough structure to prove at this general level that bisimilarity is a congruence. By instantiating the theory to concrete instances, one can then obtain congruence formats for systems such as probabilistic automata, weighted transition systems and streams—see [23] for an overview. Another advantage of abstract GSOS is that bisimulation up to context is "compatible" [34, 33], providing a sound enhancement of the bisimulation proof method which can be combined with other compatible enhancements such as bisimulation up to bisimilarity [39, 32].

Given a GSOS specification, the behaviour of terms in the syntax of the language is computed inductively, which is possible since each operator is defined directly in terms of the behaviour of its arguments. An example of a rule that does *not* fit the GSOS format is the following:

$$\frac{!x|x \xrightarrow{a} t}{!x \xrightarrow{a} t} \tag{1}$$

This rule properly defines the replication operator in CCS:[1] intuitively, $!x$ represents $x|x|x|\ldots$, i.e., the infinite parallel composition of x with itself. In fact, the above rule can be seen as assigning the behaviour of the term $!x|x$ to the simpler term $!x$, therefore we call it an *assignment rule*.

We show how to interpret assignment rules together with abstract GSOS specifications. Our approach is based on the assumption that the functor which represents the type of coalgebra is *ordered* as a complete lattice; for example, for the functor $(\mathcal{P}-)^A$ of labelled transition systems this order is simply pointwise set inclusion. The complete lattice structure gives to our disposal binary joins as well as directed ones. Binary joins are used to combine the abstract GSOS specifications with the assignment rules, whereas directed joins are needed to define the operational model on closed terms as the *least* model such that every transition can either be derived from a rule in the specification or from an assignment rule. To ensure the existence of such least models, we disallow negative premises by using *monotone* abstract GSOS specifications, a generalization of the positive GSOS format for transition systems.

The main result of this paper is that the interpretation of a monotone abstract GSOS specification together with a set of assignment rules is itself the op-

[1] The simpler rule $\frac{x \xrightarrow{a} x'}{!x \xrightarrow{a} !x|x'}$ is problematic in the presence of the sum operator, since it does not allow to derive τ-transitions from a process such as $!(a.P + \bar{a}.Q)$ [32, 41].

erational model of another (typically larger) abstract GSOS specification. Like the interpretation of a GSOS specification with assignment rules, we construct this latter specification by fixed point induction. As a direct consequence of this alternative representation of the interpretation, we obtain that bisimilarity is a congruence and that bisimulation up to context is sound and even compatible—properties that do not follow from bisimilarity being a congruence [32]. As an example, we obtain the compatibility of bisimulation up to context for CCS with replication, which was shown earlier with an ad-hoc argument (see, e.g., [32]).

In the second part of this paper, we combine *structural congruences* with the bialgebraic framework, using assignment rules. Structural congruences have been widely used in concurrency theory ever since their introduction in the operational semantics of the π-calculus in [29]. The basic idea is that SOS specifications are extended with *equations* $\equiv$ on terms, which are then linked by a special deduction rule:

$$\frac{t \equiv u \qquad u \xrightarrow{a} u' \qquad u' \equiv v}{t \xrightarrow{a} v}$$

This rule essentially states that if two processes are equated by the congruence generated by the set of equations, then they can perform the same transitions. Prototypical examples are the specification of the parallel operator by combining a single rule with commutativity, and the specification of the replication operator by an equation, both shown below:

$$\frac{x \xrightarrow{a} x'}{x|y \xrightarrow{a} x'|y} \quad x|y = y|x \qquad\qquad !x = !x|x \tag{2}$$

Even though structural congruences are standard in concurrency theory, a systematic study of their properties was missing until the work of Mousavi and Reniers, who show how to interpret SOS rules with structural congruences in various equivalent ways [30]. Mousavi and Reniers exhibit very simple examples of equations and SOS rules for which bisimilarity is not a congruence, even when the SOS rules are in the tyft (or the GSOS) format. As a solution to this problem they introduce a restricted format for equations, called cfsc (abbreviating *Congruence Format for Structural Congruences*), for which bisimilarity is a congruence when combined with tyft specifications.

In this paper, we show how to interpret structural congruences at the general level of coalgebras, in terms of an operational model on closed terms. We prove that if the equations are in the cfsc format then they can be encoded by assignment rules, in such a way that their respective interpretations coincide up to bisimilarity. Consequently, not only is bisimilarity a congruence for monotone abstract GSOS combined with cfsc equations, but we also obtain the compatibility of bisimulation up to context and bisimilarity. From a technical point of view, structural congruences have not been developed outside the work of Mousavi and Reniers, and have not at all been explored in the theory of bialgebraic semantics [3, 21]. Here, we develop the basic theory of monotone

abstract GSOS specifications for ordered functors, and use it to obtain a bialgebraic perspective on structural congruences (assuming an ordered behaviour functor).

Outline. Section 2 contains preliminaries on partial orders, (co)algebras, abstract GSOS and bisimulation(-up-to). In Section 3, assignment rules and their interpretation are introduced. We show in Section 4 that this interpretation can be obtained as the operational model of another abstract GSOS specification. Section 5 contains the integration of structural congruence with the bialgebraic framework. In Section 6, we discuss related work, and in Section 7 we conclude with some directions for future work.

To fully understand the technical development in this paper, familiarity with basic notions in category theory, bialgebraic semantics and order theory is useful. However, many of the main results and definitions are illustrated with concrete examples, in particular on the familiar case of transition systems.

2. Preliminaries

By Set we denote the category of sets and total functions. We write Id for the identity functor on Set, and $\mathsf{id}_X \colon X \to X$ or simply id for the identity function on a set X. For a relation $R \subseteq X \times Y$, we denote its left and right projections by $\pi_1 \colon R \to X$ and $\pi_2 \colon R \to Y$, respectively.

2.1. Partial Orders

Let $(P, \leq)$ be a poset. We denote the least upper bound (join) of a set $S \subseteq P$, if it exists, by $\bigvee S$. Under the assumption that they exist, we write $\bot$ for $\bigvee \emptyset$, and use the infix notation for binary joins. Note that $x \leq y$ if and only if $x \vee y = y$, for $x, y \in P$. A non-empty subset D of P is said to be *directed* if every finite subset of D has an upper bound in D. A poset P is called *directed-complete* (dcpo) if every directed subset D of P has a least upper bound $\bigvee D \in P$. A poset P is a *join semi-lattice* if every finite non-empty set has a join. If a join semi-lattice is also a dcpo, then it has all joins, and is called a complete join semi-lattice [12].

A function $f \colon P \to Q$ between two posets is *monotone* if it preserves the order, and is *continuous* if it preserves directed joins, when they exist. A morphism $f \colon P \to Q$ between two complete join semi-lattices preserves all finite and all directed joins (i.e., all joins). It follows that f is continuous, strict (i.e., $f(\bot) = \bot$), and also monotone.

For a function $f \colon P \to P$, we denote by $\mathsf{lfp}\,(f) \in P$ its *least fixed point*, if it exists. By the Knaster-Tarski theorem, if P is a complete join semilattice and f is monotone, then $\mathsf{lfp}\,(f)$ exists (e.g., [12, 40]). Slightly more generally, $\mathsf{lfp}\,(f)$ exists even if P is a dcpo with a least element $\bot$, and $f \colon P \to P$ is monotone. In this case, $\mathsf{lfp}\,(f) = f^k$ for some ordinal k, where $f^\lambda \in P$ is given, for any ordinal λ, by

$$f^0 = \bot, \qquad f^\lambda = f(\bigvee_{k<\lambda} f^k) \quad \text{for } \lambda > 0\,.$$

Note that $f^{\lambda+1} = f(f^\lambda)$ for any successor ordinal $\lambda + 1$, and that f^λ always exists because the set $\{f^k \mid k < \lambda\}$ is directed [15].

The above definition of the least fixed point of a monotone function f, as the supremum of an ascending chain, is suitable for proving properties by means of *transfinite induction*: in order to prove that a property $P(\lambda)$ holds for all ordinals λ, it is enough to prove that (a) $P(0)$ holds, the so-called base case, (b) $P(\lambda+1)$ follows from $P(\lambda)$ for any successor ordinal $\lambda+1$, and (c) $P(\lambda)$ follows from $P(k)$ for all $k < \lambda$, with λ any limit ordinal.

2.2. Signatures and Algebras

A *signature* Σ is a (possibly infinite) set of operator names $\sigma \in \Sigma$ with (finite) arities $|\sigma| \in \mathbb{N}$. To each signature Σ we associate a polynomial functor on Set, which is defined on a set X by:

$$\Sigma X = \coprod_{\sigma \in \Sigma} \{\sigma\} \times X^{|\sigma|} \tag{3}$$

Above and in the sequel we abuse notation and use Σ to represent signatures as well as their associated functors. Moreover we write $\sigma(x_1, \ldots, x_n)$ instead of $(\sigma, (x_1, \ldots, x_n))$ for elements of ΣX. The functor ΣX acts on a map $f\colon X \to Y$ as follows: $(\Sigma f)(\sigma(x_1, \ldots, x_n)) = \sigma(f(x_1), \ldots, f(x_n))$.

A Σ-*algebra* (for an arbitrary functor $\Sigma\colon \mathsf{Set} \to \mathsf{Set}$) consists of a set A and a function $\alpha\colon \Sigma A \to A$. For a signature (functor) Σ, this coincides with the standard notion of an algebra for the signature Σ: a set A together with an interpretation of every operator in Σ. A *(Σ-algebra) homomorphism* from $\alpha\colon \Sigma A \to A$ to $\beta\colon \Sigma B \to B$ is a map $f\colon A \to B$ such that $f \circ \alpha = \beta \circ \Sigma f$.

For a set of variables X and a signature Σ we denote by TX the set of *terms* over X, as defined by the grammar $t ::= \sigma(t_1, \ldots, t_n) \mid x$ where σ ranges over Σ, n is the arity of σ and x ranges over X. The special case $T\emptyset$ is the set of *closed terms*. The set of terms TX over X can be turned into a Σ-algebra $\nu_X\colon \Sigma TX \to TX$ by defining $\nu_X(\sigma(t_1, \ldots, t_n)) = \sigma(t_1, \ldots, t_n)$. This is a *free* algebra over the set X, meaning that for every Σ-algebra $\alpha\colon \Sigma A \to A$ and any function $f\colon X \to A$ there exists a unique algebra homomorphism $f^\sharp\colon TX \to A$ such that $f^\sharp(x) = f(x)$ for all $x \in X$. Intuitively, this extends a variable assignment f to terms, by using the algebra structure on A.

For every set X, we denote by $\eta_X\colon X \to TX$ the function that simply maps each $x \in X$ to itself (viewed as a term). The copairing $[\nu_X, \eta_X]\colon \Sigma TX + X \to TX$ of ν_X and η_X is an *initial* $(\Sigma + X)$-algebra. The initiality property amounts to the fact that ν_X is a free algebra, as explained above. By Lambek's lemma, each $[\nu_X, \eta_X]$ is an isomorphism [26]. In particular, for closed terms we have that $\nu_\emptyset\colon \Sigma T\emptyset \to T\emptyset$ is an isomorphism.

The definition of terms TX over a set X gives rises to a functor $T\colon \mathsf{Set} \to \mathsf{Set}$. On a function $f\colon X \to Y$, it is defined by substitution, or, more precisely, $Tf\colon TX \to TY$ is the unique homomorphism from (TX, ν_X) to (TY, ν_Y) satisfying $Tf \circ \eta_X = \eta_Y \circ f$. Every algebra $\alpha\colon \Sigma A \to A$ defines a T-algebra $\widehat{\alpha}\colon TA \to A$, as the unique homomorphism from (TA, ν_A) to (A, α) satisfying $\widehat{\alpha} \circ \eta_A = \mathsf{id}$.

Both $\eta\colon \mathsf{Id} \Rightarrow T$ and $\nu\colon \Sigma T \Rightarrow T$, with components as defined above, are natural transformations. The natural transformation ν extends to a natural transformation $\mu\colon TT \Rightarrow T$, defined on a component X by $\mu_X = \widehat{\nu_X}$, i.e., the unique homomorphism from (TTX, ν_{TX}) to (TX, ν_X) such that $\mu_X \circ \eta_{TX} = \mathsf{id}$. The triple (T, η, μ) is a monad, i.e., the following laws hold: $\mu_X \circ \eta_{TX} = \mathsf{id} = \mu_X \circ T\eta_X$ and $\mu_X \circ T\mu_X = \mu_X \circ \mu_{TX}$. It is the *free monad* on Σ, see [2].

2.3. Coalgebras

For an extensive treatment with many examples we refer to [37, 19]. Let $F\colon \mathsf{Set} \to \mathsf{Set}$ be a functor. An (F-)coalgebra is a pair (X, α) where X is a set (of states) and $\alpha\colon X \to FX$ is a function. Let (X, α) and (Y, β) be two coalgebras. A function $f\colon X \to Y$ is an *(F-coalgebra) homomorphism* if $Ff \circ \alpha = \beta \circ f$.

Example 2.1. *Labelled transition systems* (LTSs) over a set of labels A are coalgebras for the functor $FX = (\mathcal{P}X)^A$. For an LTS $\alpha\colon X \to (\mathcal{P}X)^A$ we write $x \xrightarrow{a} x'$ iff $x' \in \alpha(x)(a)$. Intuitively, for a state $x \in X$, $\alpha(x)(a)$ contains all the outgoing transitions from x labelled by a. *Image-finite* labelled transition systems are coalgebras for the functor $FX = (\mathcal{P}_\omega X)^A$, where $\mathcal{P}_\omega$ is the finite power set functor, mapping a set X to the set of finite subsets of X.

Weighted transition systems for a set of labels A and a complete monoid M (i.e., a monoid with an infinitary sum operation consistent with the finite sum [10]) are coalgebras for the functor $(M^-)^A$ where $M^-\colon \mathsf{Set} \to \mathsf{Set}$ is defined as follows:

- For each set X, M^X is the set of functions from X to M.
- For each function $h\colon X \to Y$, $M^h\colon M^X \to M^Y$ is the function mapping each $\varphi \in M^X$ to $\varphi^h \in M^Y$ defined, for all $y \in Y$, by

$$\varphi^h(y) = \sum_{x' \in h^{-1}(y)} \varphi(x')\,.$$

Given a weighted transition system $\alpha\colon X \to (M^X)^A$, we write $x \xrightarrow{a,r} y$ if $\alpha(x)(a)(y) = r$ and $r \neq 0$. Note that the above definition allows infinitely branching weighted transition systems.

By taking the Boolean monoid we retrieve labelled transition systems. More generally, every complete join-semilattice is a complete (idempotent) monoid, giving us a large source of interesting weighted transition systems. For example, the set of all subsets of given a set S forms a complete monoid under either union or intersection operations; similarly for the set of all languages over an alphabet S.

Another example that we will consider later in Section 3 is given by weighted transition systems over the set $M = \mathbb{R}^+ \cup \{\infty\}$ of positive reals, ordered as usual and extended with a top element ∞. Together with the supremum operation, M forms a complete join semi-lattice, and hence a complete monoid with 0 as

unit (the least element). Similar examples can be found by taking $[0,1]$ as base set instead of $\mathbb{R}^+ \cup \{\infty\}$. For many more examples of complete monoids we refer to [10].

We recall the notion of bisimulation between coalgebras based on relation lifting [14, 36]. Given a functor F, the *relation lifting* Rel_F maps a relation $R \subseteq X \times Y$ to

$$\mathsf{Rel}_F(R) = \{(b_1, b_2) \in FX \times FY \mid \exists z \in FR \text{ s.t. } F\pi_1(z) = b_1 \text{ and } F\pi_2(z) = b_2\}\,.$$

Given two F-coalgebras (X, α) and (Y, β), define $b_{\alpha,\beta} \colon \mathcal{P}(X \times Y) \to \mathcal{P}(X \times Y)$ by

$$b_{\alpha,\beta}(R) = \{(x, y) \mid (\alpha(x), \beta(y)) \in \mathsf{Rel}_F(R)\}\,.$$

A relation $R \subseteq X \times Y$ is a *bisimulation* (between α and β) if $R \subseteq b_{\alpha,\beta}(R)$. The greatest bisimulation is called *bisimilarity* and is denoted by $\sim_{\alpha,\beta}$. If $\alpha = \beta$ then we write b_α and $\sim_\alpha$ instead of $b_{\alpha,\beta}$ and $\sim_{\alpha,\beta}$, respectively.

2.4. Bisimulation up-to

To state and prove the applications of our main results to the proof technique of bisimulation-up-to, we recall here a few of the elements of its (coalgebraic) theory from [32, 33]. We only mention those definitions and results that are strictly necessary, and refer the reader to the aforementioned papers for the full theory, motivation and examples. Most of the technical development in the paper does not depend on the definitions below, and thus can safely be skipped by the reader.

Let $f, g \colon \mathcal{P}(X \times Y) \to \mathcal{P}(X \times Y)$ be functions that are monotone (with respect to the inclusion order). We say f is *g-compatible* if for any two relations R and S: $R \subseteq g(S)$ implies $f(R) \subseteq g(f(S))$. Compatibility implies soundness, that is, if $R \subseteq g(f(R))$, then $R \subseteq S$ for some relation S satisfying $S \subseteq g(S)$ [32].

In this paper, we are interested only in instantiating g to $b_{\alpha,\beta}$, and we write compatibility (on α, β) instead of $b_{\alpha,\beta}$-compatibility. A relation R for which $R \subseteq b_{\alpha,\beta}(f(R))$ is called a *bisimulation up to f*. If f is compatible, establishing that R is a bisimulation up to f suffices to prove that R is contained in bisimilarity.

We will be interested in two instantiations of f above, both defined on a single coalgebra $\alpha \colon X \to FX$. Let $bis(R) = \sim_\alpha \circ R \circ \sim_\alpha$; a bisimulation up to *bis* is called a *bisimulation up to bisimilarity*. Further, if there is an algebra $\beta \colon \Sigma X \to X$ we can define the contextual closure function $ctx_\beta(R) = \{(\beta(t), \beta(u)) \mid (t, u) \in \mathsf{Rel}_\Sigma(R)\}$; a bisimulation up to ctx_β is also called a *bisimulation up to context*. If β is clear from the context, we write ctx instead of ctx_β. Bisimulation up to bisimilarity is compatible on any coalgebra for a functor that preserves weak pullbacks, and bisimulation up to context is compatible if (X, α, β) is a λ-bialgebra for some distributive law of Σ over F, see [33] for both compatibility results.

In Section 5, we will prove the compatibility of the combined up-to technique $bis \circ ctx \circ bis$. Spelling out the details, this up-to technique maps a relation R to $\sim \circ\, ctx(\sim \circ R \circ \sim) \circ \sim$, where $\sim$ is the bisimilarity relation on the coalgebra under

consideration. The reason for considering this up-to technique in Theorem 5.13 is that we are not able to prove compatibility of ctx directly in the context of that result. However, for every relation R, we have $ctx(R) \subseteq bis \circ ctx \circ bis(R)$, so that the latter is at least as useful as ctx, as an up-to technique.

2.5. Equality up to bisimilarity

For our main result of Section 5, we introduce a strong notion of equivalence between coalgebras on the same carrier, intuitively capturing that the two coalgebras behave the same up to bisimilarity. We prove certain basic facts about this notion; these can be safely skipped by the reader, as they are only required in the proof of Theorem 5.13.

Definition 2.2. Let $\alpha, \beta \colon X \to FX$ be F-coalgebras on a common carrier X. We say α and β are *equal up to bisimilarity* if the bisimilarity relation $\sim_{\alpha,\beta}$ between α and β is reflexive.

If F preserves weak pullbacks, then an equivalent definition is that the identity relation Δ is a bisimulation up to bisimilarity.

Lemma 2.3. *Let $\alpha, \beta \colon X \to FX$ be coalgebras that are equal up to bisimilarity and assume that F preserves weak pullbacks. Then $\sim_\alpha = \sim_{\alpha,\beta} = \sim_\beta$.*

Proof. Since F preserves weak pullbacks, by [37, Theorem 5.4], the composition of two bisimulations is again a bisimulation. Further, by assumption, $\sim_{\alpha,\beta}$ is reflexive. We prove $\sim_\alpha = \sim_{\alpha,\beta}$; the equality $\sim_{\alpha,\beta} = \sim_\beta$ is similar.

We have $\sim_\alpha \subseteq \sim_\alpha \circ \sim_{\alpha,\beta}$ by reflexivity of $\sim_{\alpha,\beta}$, and $\sim_\alpha \circ \sim_{\alpha,\beta} \subseteq \sim_{\alpha,\beta}$ since $\sim_\alpha \circ \sim_{\alpha,\beta}$ is a bisimulation between α and β and therefore contained in $\sim_{\alpha,\beta}$, the greatest such bisimulation.

For the converse inclusion, we use that the inverse relation $\sim_{\alpha,\beta}^{-1}$ is a bisimulation between β and α (see [37, Theorem 5.2]), so that the composition $\sim_{\alpha,\beta} \circ \sim_{\alpha,\beta}^{-1}$ is a bisimulation on α, hence $\sim_{\alpha,\beta} \circ \sim_{\alpha,\beta}^{-1} \subseteq \sim_\alpha$. Since $\sim_{\alpha,\beta}^{-1}$ is reflexive, we obtain $\sim_{\alpha,\beta} \subseteq \sim_{\alpha,\beta} \circ \sim_{\alpha,\beta}^{-1} \subseteq \sim_\alpha$. □

In the following we will use a standard result about relation lifting: if the functor F preserves weak pullbacks, then for any relations R, S we have $\mathsf{Rel}_F(R) \circ \mathsf{Rel}_F(S) = \mathsf{Rel}_F(R \circ S)$ (see, e.g., [18]).

Lemma 2.4. *Let F, α and β be as in Lemma 2.3.*

(a) If $R \subseteq b_\alpha(S)$ then $bis(R) \subseteq b_\beta(bis(S))$.
(b) If f is b_β-compatible then $bis \circ f \circ bis$ is b_α-compatible.

where bis is defined w.r.t. the bisimilarity relation $\sim$ (of both α and β).

Proof. (a) Suppose $R \subseteq b_\alpha(S)$, and let $(x, y) \in R$; then $\alpha(x)\, \mathsf{Rel}(F)(S)\, \alpha(y)$. Since α and β are equal up to bisimilarity, we have $\beta(x)\, \mathsf{Rel}(F)(\sim)\, \alpha(x)$ and $\alpha(y)\, \mathsf{Rel}(F)(\sim)\, \beta(y)$. Hence

$$\beta(x)\, \mathsf{Rel}(F)(\sim)\, \alpha(x)\, \mathsf{Rel}(F)(S)\, \alpha(y)\, \mathsf{Rel}(F)(\sim)\, \beta(y)\,.$$

Since F preserves weak pullbacks, this implies $\beta(x)\,\mathsf{Rel}(F)(\sim \circ S \circ \sim)\,\beta(y)$. Thus $R \subseteq b_\beta(\sim \circ S \circ \sim)$; by compatibility of bis this implies $\sim \circ R \circ \sim \subseteq b_\beta(\sim \circ \sim \circ S \circ \sim \circ \sim)$, and by transitivity of $\sim$ (F preserves weak pullbacks) then $bis(R) \subseteq b_\beta(bis(S))$.

(b) Suppose $R \subseteq b_\alpha(S)$. By (i) we get $bis(R) \subseteq b_\beta(bis(S))$. We apply b_β-compatibility of f to obtain $f \circ bis(R) \subseteq b_\beta(f \circ bis(S))$. Finally, again by (i) (replacing α by β and vice versa) we get $bis \circ f \circ bis(R) \subseteq b_\alpha(bis \circ f \circ bis(S))$. □

2.6. Bialgebraic Operational Semantics

See [23] for an overview of this topic. In the remainder of this paper, we assume a fixed signature Σ with associated free monad T, and a Set endofunctor F representing the type of behaviour. An *(abstract GSOS) specification* is a natural transformation of the form

$$\rho\colon \Sigma(F \times \mathsf{Id}) \Rightarrow FT\,.$$

As first observed by Turi and Plotkin [42], if F is the functor $(\mathcal{P}_\omega -)^A$ of image-finite labelled transition systems then specifications of the above type can be induced by specifications in the well-known GSOS format, introduced in [4]. A GSOS rule for an operator $\sigma \in \Sigma$ of arity n is of the form

$$\frac{\{x_{i_j} \xrightarrow{a_j} y_j\}_{j=1..m} \qquad \{x_{i_k} \overset{b_k}{\not\to}\}_{k=1..l}}{\sigma(x_1,\ldots,x_n) \xrightarrow{c} t} \tag{4}$$

where m is the number of positive premises, l is the number of negative premises, and $a_1,\ldots,a_m,b_1,\ldots,b_l,c \in A$ are labels. The variables $x_1,\ldots,x_n$, $y_1,\ldots,y_m$ are pairwise distinct, and t is a term over these variables.

Example 2.5. As a running example, we consider a very basic process calculus, which has a constant $\mathsf{0}$, a unary prefix operator $a.x$ for each $a \in A$ with A some fixed set of labels, and a binary parallel operator $x|y$. The GSOS rules are as follows:

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{x|y \xrightarrow{a} x'|y} \qquad \frac{y \xrightarrow{a} y'}{x|y \xrightarrow{a} x|y'}$$

For simplicity of the presentation, the parallel operator does not have a rule for synchronization.

We present the syntax as a functor $\Sigma X = X \times X + A \times X + 1$, and write $x|y$, $a.x$ and $\mathsf{0}$ respectively for elements of the three disjoint sets in ΣX. The above GSOS specification corresponds to the abstract GSOS specification $\rho\colon \Sigma(F \times \mathsf{Id}) \Rightarrow FT$, where $FX = (\mathcal{P}X)^A$ and T is the free monad on Σ, defined on a component X as follows, by cases on the operators in the signature:

$$\begin{aligned}
\rho_X(\mathsf{0}) &= \lambda a.\emptyset \\
\rho_X(a.(f,x)) &= \lambda b.\begin{cases}\{x\} & \text{if } a = b \\ \emptyset & \text{otherwise}\end{cases} \\
\rho_X((f,x)|(g,y)) &= \lambda a.\{x'|y \mid x' \in f(x)\} \cup \{x|y' \mid y' \in g(x)\}
\end{aligned}$$

for all $(f, x), (g, y) \in (\mathcal{P}X)^A \times X$.

If we instantiate F to the functor $\mathbb{R} \times \mathsf{Id}$ of stream systems over the reals, specifications correspond to the format of *behavioural differential equations* [38] presented in [25]. By instantiating abstract GSOS specifications to other functors one can obtain formats for many types of systems, including, e.g., syntactic formats for probabilistic and weighted transition systems [3, 22].

Each specification $\rho\colon \Sigma(F \times \mathsf{Id}) \Rightarrow FT$ induces a unique coalgebra

$$M(\rho)\colon T\emptyset \to FT\emptyset$$

with the following property:

$$M(\rho) \circ \nu_\emptyset = F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle M(\rho), \mathsf{id}\rangle \;:\; \Sigma T\emptyset \to FT\emptyset\,. \tag{5}$$

We call this coalgebra $M(\rho)$ the *operational model*, or also the *ρ-model* on the initial algebra $(T\emptyset, \nu_\emptyset)$.

By Equation (5), to compute the behaviour of a term $\sigma(t_1, \ldots, t_n)$ in $M(\rho)$, we may compute the behaviour of its subterms $t_1, \ldots, t_n$ and then instantiate a rule from ρ. For labelled transition systems, $M(\rho)$ is precisely the unique *supported model* corresponding to a GSOS specification ρ: every transition in f is derived from rules in the specification ρ and each derivable transition occurs in f (see [1, 4]).

An important property of GSOS is that bisimilarity is a congruence on the operational model corresponding to a specification [4]. Turi and Plotkin [42] proved at the general level of abstract GSOS specifications that behavioural equivalence on the operational model is a congruence, extending the result of [4] from labelled transition systems to arbitrary types of coalgebras.

Any abstract GSOS specification ρ can be extended to a natural transformation $\rho^*\colon T(F \times \mathsf{Id}) \Rightarrow (F \times \mathsf{Id})T$, which is a *distributive law* of the monad T over the cofree copointed functor $F \times \mathsf{Id}$. The latter means that ρ^* satisfies certain laws, which however we do not need to recall here. The construction of ρ^* from ρ is well explained in, e.g., [3, Lemma 3.5.2]; we recall the basics here for convenience of the reader. On a component X, it is defined as the unique map ρ^*_X making the following diagram commute:

$$\begin{array}{ccc}
\Sigma T(FX \times X) & \xrightarrow{\Sigma\rho^*_X} & \Sigma(FTX \times TX) \\
 & & \downarrow{\scriptstyle \langle \rho_{TX}, \Sigma\pi_2\rangle} \\
\downarrow{\scriptstyle \nu_{FX\times X}} & & FTTX \times \Sigma TX \\
 & & \downarrow{\scriptstyle F\mu_X \times \nu_X} \\
T(FX \times X) & \xrightarrow{\rho^*_X} & FTX \times TX \\
\uparrow{\scriptstyle \eta_{FX\times X}} & \nearrow{\scriptstyle F\eta_X \times \eta_X} & \\
FX \times X & &
\end{array} \tag{6}$$

Actually, the standard definition as in [3] instead uses the algebra $F\mu_X \times \mu_X \circ \langle \rho_{TX}, \kappa_{TX} \circ \Sigma\pi_2 \rangle$ where $\kappa \colon \Sigma \Rightarrow T$ is the canonical embedding, but this is equivalent to the above, since $\mu \circ \kappa_T = \nu$ (see [3, Lemma 2.2.8]). The existence and uniqueness of ρ^*_X is justified by the fact that $\nu_{FX\times X}$ is a free algebra. We need the following general property of abstract GSOS specifications: $\rho^* \circ \kappa_{F\times \mathsf{Id}} = \langle \rho_X, \kappa \circ \Sigma\pi_2 \rangle$, which follows from (the proof of) [3, Lemma 3.4.24]. Concretely, for Σ a polynomial functor defined from a signature, it means that for any operator σ of arity n and any $(f_1, x_1), \ldots, (f_n, x_n) \in FX \times X$:

$$\rho^*_X(\sigma((f_1, x_1), \ldots, (f_n, x_n))) = (\rho_X((f_1, x_1), \ldots, (f_n, x_n)), \sigma(x_1, \ldots, x_n)) \,. \quad (7)$$

For the operational model $M(\rho) \colon T\emptyset \to FT\emptyset$ we have the following property:

$$\langle M(\rho), \mathsf{id} \rangle \circ \mu_\emptyset = (F\mu_\emptyset \times \mu_\emptyset) \circ (\rho^*)_{T\emptyset} \circ T\langle M(\rho), \mathsf{id} \rangle \,. \quad (8)$$

which means that the triple $(T\emptyset, \mu_\emptyset, \langle M(\rho), \mathsf{id} \rangle)$ is a so-called ρ^*-*bialgebra*. In fact, more generally there is a one-to-one correspondence between models of ρ and ρ^*-bialgebras. However, a more detailed discussion is not needed in this paper, and we only need to know that the operational model satisfies Equation (8) for certain proofs. Finally, following the discussion at the end of Section 2.4, for this bialgebra, we note that the contextual closure $ctx_{\mu_\emptyset}$ is $b_{\langle M(\rho), \mathsf{id} \rangle}$-compatible. A careful explanation of the meaning of that result is beyond the scope of this paper; instead, we refer to [33] for details.

3. Adding Assignment Rules

In this section we consider the interpretation of abstract GSOS specifications (without negative premises) together with assignments of the form

$$\sigma(x_1, \ldots, x_n) := t \quad (9)$$

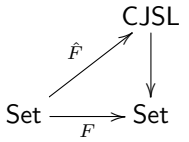
where t is a term over the variables $x_1, \ldots, x_n$. We call these *assignment rules*; they are defined more formally below in Definition 3.6. Assignment rules will be interpreted as a kind of rewriting rules: the behaviour of t induces the behaviour of $\sigma(x_1, \ldots, x_n)$. An example is the replication operator given in Equation (1) of the introduction; this can be given by the assignment rule $!x := !x|x$. Notice that assignment rules do not fit directly into the bialgebraic framework, since they are inherently non-structural: they do not satisfy the property of GSOS specifications that the behaviour of terms in the operational model is computed directly from the behaviour of their subterms.

In the case of labelled transition systems, given a GSOS specification and a set of rules of the above form, the desired interpretation is informally as follows (this is formalized below): every transition from a term $\sigma(t_1, \ldots, t_n)$ should either be derived from the transitions of $t_1, \ldots, t_n$ and a rule in the specification, or from an assignment rule which has σ on the left-hand side. However, such an interpretation is not necessarily unique, since there may be infinite inferences caused by the assignment rules. For example, the rule $\sigma(x) := \sigma(x)$ does not

have a unique solution. In order to rule out such cases, one is interested in the *least* transition system on closed terms which is a model in the above sense. In the presence of assignment rules, such a least model does not necessarily exist in general because of negative premises. Therefore, we will restrict to positive GSOS specifications, which do not feature negative premises. (We mention that there are different ways of dealing with negative premises [13], but throughout this paper we simply avoid negative premises altogether.)

We interpret specifications which involve assignment rules at the general level of a functor F based on a fixed point construction, using the assumption that F is ordered as a complete lattice. In the case of labelled transition systems this order is clear and often left implicit: the order on $FX = (\mathcal{P}X)^A$ is given by (pointwise) subset inclusion.

For the general case, we assume that our behaviour functor F is *ordered* [16], i.e., factors through CJSL—the category of complete (join semi-)lattices and join-preserving functions. That is, we assume a functor $\hat{F}\colon \mathsf{Set} \to \mathsf{CJSL}$ such that the following triangle commutes:



where the arrow from CJSL to Set is the forgetful functor, which takes a complete lattice to its underlying set. Basically a functor F is ordered if and only if, for any set X, the set FX can be enriched with a complete join semi-lattice structure, and, moreover, for any function $f\colon X \to Y$, Ff is join-preserving (w.r.t. the join semi-lattice structure). Consequently, Ff is also monotone, i.e., for any $x, y \in FX$: $x \leq y$ implies $(Ff)(x) \leq (Ff)(y)$.

Example 3.1. As mentioned above, the functor $(\mathcal{P}-)^A$ of labelled transition systems has a natural complete join semi-lattice structure given by the pointwise extension of subset inclusion. Moreover, for any set of functions $\{f_i\colon A \to \mathcal{P}(X)\}_{i\in I}$, function $h\colon X \to Y$, and $a \in A$, we have that

$$\begin{array}{rcl} ((\mathcal{P}h)^A(\bigvee_I f_i))(a) & = & \{h(x) \mid x \in (\bigvee f_i)(a)\} \\ & = & \{h(x) \mid x \in \bigcup_I f_i(a)\} \\ & = & \bigcup_I \{h(x) \mid x \in f_i(a)\} \\ & = & \bigcup_I ((\mathcal{P}h)^A(f_i))(a) \\ & = & (\bigvee_I (\mathcal{P}h)^A(f_i))(a)\,. \end{array}$$

It follows that $(\mathcal{P}-)^A$ is an ordered functor.

Generalizing the previous example of labelled transition systems, if F is an ordered functor then so is F^A, with a complete join semi-lattice structure given by pointwise extension. Furthermore, if F and G are ordered functors then so is the functor $F \times G$, with a complete join semi-lattice structure given by the natural pairwise order.

Every complete join-semilattice M is a complete monoid with infinitary sum as supremum. Next we show that in this case the functor $M^-\colon \mathsf{Set} \to \mathsf{Set}$ is ordered. For every set X, because M is a complete join semi-lattice, so is M^X, where the order is given by pointwise extension. Moreover, for any function $h\colon X \to Y$, M^h preserves arbitrary joins. To see this, let $\{\varphi_i\colon X \to M\}_{i\in I}$ and $y \in Y$. We have:

$$\begin{array}{rcl} M^h(\bigvee_I \varphi_i)(y) & = & \sum_{x\in h^{-1}(y)}(\bigvee_I \varphi_i)(x) \\ & = & \sum_{x\in h^{-1}(y)} \sum_I \varphi_i(x) \\ & = & \sum_I \sum_{x\in h^{-1}(y)} \varphi_i(x) \\ & = & \sum_I M^h(\varphi_i)(y) \\ & = & (\bigvee_I M^h(\varphi_i))(y)\,. \end{array}$$

Note that every complete monoid is necessarily commutative [10], but it does not need to be idempotent. It follows that not every complete monoid is a complete join-semilattice. For example, any subset $\{0,\dots,k\}$ of natural numbers equipped with addition truncated at k is a non-idempotent complete monoid with 0 as unit, but it is not a complete join-semilattice.

Example 3.2. Consider the set $M = \mathbb{R}^+ \cup \{\infty\}$ of positive reals, ordered as usual and extended with a top element ∞. Together with the usual supremum operation M is a complete join semi-lattice, and hence a complete monoid. By the above, it extends to an order on the functor for weighted transition systems over this monoid, where joins are calculated pointwise. Similarly, the functor for weighted automata $M \times (M^-)^A$ is an ordered functor.

Example 3.3. One can be tempted to extend any functor $F\colon \mathsf{Set} \to \mathsf{Set}$ to a CJSL-ordered functor F' by defining $F'X = FX + 2$, using the discrete order on FX and taking the elements of $2 = \{\top, \bot\}$ to be the top and the bottom element respectively. However, contrary to what is stated in [35, Example 2], such a functor F' is not CJSL-ordered, in general. Indeed $F'X$ is a complete join semi-lattice, but the functor F' is not ordered because $F'f$ is not necessarily join-preserving, for a function f. For instance, if we take $F = \mathsf{Id}$, a set X with two distinct elements $x, y \in X$ and a function $f\colon X \to X$ such that $f(x) = f(y)$, we have $(F'f)(x) \vee (F'f)(y) = f(x) \vee f(y) = f(x) \neq \top$ whereas $(F'f)(x \vee y) = (F'f)(\top) = \top$.

Given arbitrary sets X and Y, the complete lattice on FY lifts pointwise to a complete lattice on functions of type $X \to FY$, i.e., for a collection $\{f_i\}_{i\in I}$ of functions of the form $f_i\colon X \to FY$ we define $(\bigvee\{f_i\}_{i\in I})(x) = \bigvee_{i\in I}(f_i(x))$. This induces in particular a complete lattice on the set of all coalgebras on closed terms, which we denote by

$$\mathbb{M} = \{f \mid f\colon T\emptyset \to FT\emptyset\}\,.$$

The order on F lifts to an order on $F \times \mathsf{Id}$ by defining $(b_1, x_1) \leq (b_2, x_2)$ iff $b_1 \leq b_2$ and $x_1 = x_2$ for $(b_1, x_1), (b_2, x_2) \in FX \times X$. Moreover, the order lifts component-wise to ΣFX (and also to $\Sigma(FX \times X)$) for any set X, by defining,

for any $\sigma, \tau \in \Sigma$ of arity n and m respectively, $\sigma(k_1, \ldots, k_n) \leq \tau(l_1, \ldots, l_m)$ iff $\sigma = \tau$ (so also $n = m$) and $k_i \leq l_i$ for all $i \leq n$.

Definition 3.4. Using the above lifting of the order on F to $\Sigma(F \times \mathsf{Id})$, a specification $\rho\colon \Sigma(F \times \mathsf{Id}) \Rightarrow FT$ is said to be *monotone* if all its components are.

We refer to [5, 11] for a more general account of monotone abstract GSOS.

Example 3.5. As stated in [11], for the functor $F = (\mathcal{P}-)^A$ of labelled transition systems, monotone specifications correspond to specifications in (an infinitary version of) the *positive GSOS* format.

Assignment rules (9) can be formalised categorically in terms of natural transformations. These are independent of the behaviour functor F.

Definition 3.6. An *assignment rule* is a natural transformation $d\colon \Sigma \Rightarrow T$.

If there is no intended assignment for an operator $\sigma \in \Sigma$, this is modelled by defining $d_X(\sigma(x_1, \ldots, x_n)) = \sigma(x_1, \ldots, x_n)$ for every X and $x_1, \ldots, x_n \in X$.

Example 3.7. Recall the syntax of Example 2.5, i.e., a constant $\mathsf{0}$, a unary operator $a.x$ for each $a \in A$ and a binary operator $x|y$. We extend this with a unary operator $!x$, and we model this extended syntax by the functor $\Sigma X = X \times X + X + A \times X + 1$. Following Example 2.5, we denote an element x of the set X in ΣX by $!x$.

Consider the assignments $x|y := y|x$ and $!x := !x|x$. These are modelled formally by an assignment rule $d\colon \Sigma \Rightarrow T$, defined on a component X by cases on the operators in the signature:

$$d_X(\mathsf{0}) = \mathsf{0} \qquad d_X(a.x) = a.x \qquad d_X(x|y) = y|x \qquad d_X(!x) = !x|x\,. \tag{10}$$

The above assignment rule formalizes both assignments at once; they could also be defined by two separate assignment rules. Notice that $\mathsf{0}$ and $a.x$ are mapped simply to themselves, reflecting that there are no assignments for these operators.

In the above example we only needed a single assignment rule. To allow multiple assignments for a single operator, we will work in the remainder with a set of assignment rules.

Assumption 3.8. In the remainder of this paper we assume:

1. A CJSL-ordered functor F.
2. A functor Σ defined from a signature (see Section 2.2), with free monad (T, η, μ).
3. A monotone GSOS specification $\rho\colon \Sigma(F \times \mathsf{Id}) \Rightarrow FT$.
4. A set Δ of assignment rules, ranged over by $d\colon \Sigma \Rightarrow T$.

Now we have all the necessary tools to define a model on closed terms of an abstract GSOS specification together with a set of assignment rules. Our definition extends Equation (5) in Section 2.6, which characterizes the operational model of a GSOS specification ρ, by incorporating a set of assignment rules.

Definition 3.9. Let $\psi\colon \mathbb{M} \to \mathbb{M}$ be the (unique) function such that

$$\psi(f) \circ \nu_\emptyset = F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{d\in\Delta} f \circ \mu_\emptyset \circ d_{T\emptyset} \;:\; \Sigma T\emptyset \to FT\emptyset\,.$$

A (ρ, Δ)-model is a coalgebra $f \in \mathbb{M}$ such that $\psi(f) = f$.

The function ψ is indeed uniquely defined, since $\nu_\emptyset\colon \Sigma T\emptyset \to T\emptyset$ is an initial algebra and therefore an isomorphism (Section 2.2). As argued in the beginning of this section, in general there may be more than one model for a fixed ρ and Δ, and we regard the *least* (ρ, Δ)-model to be the correct interpretation. In order to show that a least model exists we need the following.

Lemma 3.10. *The function $\psi\colon \mathbb{M} \to \mathbb{M}$ is monotone.*

Proof. Suppose $f \leq g$ for some $f, g \in \mathbb{M}$. Then by monotonicity of ρ we have $\rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq \rho_{T\emptyset} \circ \Sigma\langle g, \mathsf{id}\rangle$, and since $F\mu_\emptyset$ is monotone then $F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle g, \mathsf{id}\rangle$. It follows that $\psi(f) \circ \nu_\emptyset \leq \psi(g) \circ \nu_\emptyset$ and thus also $\psi(f) \leq \psi(g)$ because $\nu_\emptyset$ is an isomorphism. □

Since ψ is monotone and $\mathbb{M}$ is a complete lattice, by the Knaster-Tarski theorem, ψ has a least fixed point.

Definition 3.11. The *interpretation* of ρ and Δ is the least (ρ, Δ)-model.

Example 3.12. Informally, for a GSOS specification together with assignment rules, the interpretation is the least transition system on closed terms so that $\sigma(t_1, \ldots, t_n) \xrightarrow{a} t'$ if and only if:

1. it can be obtained by instantiating a rule in the specification, or
2. there is an assignment of t to σ, and $t \xrightarrow{a} t'$.

As a concrete example, consider the following specification with assignment rules.

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{x|y \xrightarrow{a} x'|y} \qquad x|y := y|x \qquad !x := !x|x \tag{11}$$

The relevant syntax (with the operator $\mathsf{0}$) and the assignments are modelled respectively by the functor Σ and the assignment rule d from Example 3.7.

Notice that, contrary to Example 2.5, for the parallel operator we only have one of the two rules in the above specification. The GSOS rules in the above specification correspond to a natural transformation $\rho\colon \Sigma(F \times \mathsf{Id}) \Rightarrow FT$, where $FX = (\mathcal{P}X)^A$, defined on a component X as follows, for all $(f, x), (g, y) \in (\mathcal{P}X)^A \times X$:

$$\rho_X((f, x)|(g, y)) = \lambda a.\{x'|y \mid x' \in f(x)\}$$
$$\rho_X(!(f, x)) = \lambda a.\emptyset$$

and with $\rho_X(\mathsf{0})$,$\rho_X(a.(f,x))$ as in Example 2.5. The definition of $\rho_X(!(f,x))$ assigns the least element of FTX since there are no GSOS rules for $!x$.

We now compute the interpretation of $(\rho, \{d\})$, for the above ρ and d corresponding to the specification (11). First, we spell out $\psi(f)$ for a given coalgebra $f\colon T\emptyset \to FT\emptyset$. For any $a \in A$ and $t, u \in T\emptyset$:

$$\begin{aligned}\psi(f)(\mathsf{0}) &= F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle(\mathsf{0}) \vee f \circ \mu_\emptyset \circ d_\emptyset(\mathsf{0}) \\ &= F\mu_\emptyset \circ \rho_{T\emptyset}(\mathsf{0}) \vee f(\mathsf{0}) \\ &= (\lambda a.\emptyset) \vee f(\mathsf{0}) \\ &= f(\mathsf{0})\end{aligned}$$

$$\begin{aligned}\psi(f)(a.t) &= F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle(a.t) \vee f \circ \mu_\emptyset \circ d_\emptyset(a.t) \\ &= F\mu_\emptyset \circ \rho_{T\emptyset}(a.(f(t), t)) \vee f(a.t) \\ &= \left(\lambda b. \begin{cases} \{t\} & \text{if } a = b \\ \emptyset & \text{otherwise} \end{cases}\right) \vee f(a.t)\end{aligned}$$

$$\begin{aligned}\psi(f)(t|u) &= F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle(t|u) \vee f \circ \mu_\emptyset \circ d_\emptyset(t|u) \\ &= F\mu_\emptyset \circ \rho_{T\emptyset}((f(t), t)|(f(u), u)) \vee f(u|t) \\ &= (\lambda a.\{t'|u \mid t' \in f(t)(a)\}) \vee f(u|t)\end{aligned}$$

$$\begin{aligned}\psi(f)(!t) &= F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle(!t) \vee f \circ \mu_\emptyset \circ d_\emptyset(!t) \\ &= F\mu_\emptyset \circ \rho_{T\emptyset}(!(f(t), t)) \vee f(!t|t) \\ &= (\lambda a.\emptyset) \vee f(!t|t) \\ &= f(!t|t)\end{aligned}$$

Now recall that the interpretation of $(\rho, \{d\})$ is defined as the least fixed point of the function ψ given in Definition 3.9, which is also the least pre-fixed point, i.e., the least f such that $\psi(f) \leq f$. By the above computations, this means that the interpretation f is the least transition system such that for all $a \in A$ and $t, u \in T\emptyset$:

- $t \in f(a.t)(a)$,
- $\{t'|u \mid t' \in f(t)(a)\} \subseteq f(t|u)(a)$,
- $f(u|t)(a) \subseteq f(t|u)(a)$,
- $f(!t|t)(a) \subseteq f(!t)(a)$.

This is the desired interpretation of the specification in (11), as the least transition system satisfying both the GSOS rules and the assignments.

4. Abstract GSOS Specifications for Assignment Rules

In the previous section, we have seen how to interpret an abstract GSOS specification ρ together with a set of assignment rules Δ as a coalgebra on closed terms. In this section, we show that we can alternatively construct this coalgebra as the operational model of another specification (without assignment rules), which is constructed as a least fixed point of a function on the complete lattice of specifications. The consequence of this alternative representation is that the well-behavedness properties of the operational model of a specification, such as bisimilarity being a congruence and the compatibility of bisimulation up to context, carry over to the interpretation of ρ and Δ.

Let $\mathbb{S}$ be the set of all monotone abstract GSOS specifications (Definition 3.4). We turn $\mathbb{S}$ into a complete lattice by defining the order componentwise, i.e., for any $L \subseteq \mathbb{S}$ and any set X: $(\bigvee L)_X = \bigvee_{\rho \in L} \rho_X$. The join is well-defined:

Lemma 4.1. *For any $L \subseteq \mathbb{S}$: the family of functions $\bigvee L$ as defined above is a monotone specification.*

Proof. Let $f\colon X \to Y$ be a function. For any $k \in \Sigma(FX \times X)$:

$$\begin{array}{lll} FTf \circ (\bigvee L)_X(k) & = FTf \circ (\bigvee_{\rho \in L}(\rho_X(k))) & \text{definition of } \bigvee L \\ & = \bigvee_{\rho \in L}(FTf \circ \rho_X(k)) & FTf \text{ is join-preserving} \\ & = \bigvee_{\rho \in L}(\rho_Y \circ \Sigma(Ff \times f)(k)) & \text{naturality of } \rho \\ & = (\bigvee L)_Y(\Sigma(Ff \times f)(k)) & \text{definition of } \bigvee L \end{array}$$

which proves naturality.

For monotonicity, let $k, l \in \Sigma(FX \times X)$ with $k \leq l$. For each $\rho \in L$ we have that ρ is monotone, so that $\rho_X(k) \leq \rho_X(l)$. Hence $\bigvee_{\rho \in L}(\rho_X(k)) \leq \bigvee_{\rho \in L}(\rho_X(l))$. We thus obtain

$$(\bigvee L)(k) = \bigvee_{\rho \in L}(\rho_X(k)) \leq \bigvee_{\rho \in L}(\rho_X(l)) = (\bigvee L)(l)$$

as desired. □

The lattice structure of $\mathbb{S}$ provides a way of combining specifications. Next, we will use the lattice structure of $\mathbb{S}$ in the definition of a function on $\mathbb{S}$. That function is based on natural transformations of the following form, defined from a given assignment rule $d \in \Delta$ and specification τ:

$$\Sigma(F \times \mathsf{Id}) \xrightarrow{d_{F \times \mathsf{Id}}} T(F \times \mathsf{Id}) \xrightarrow{\tau^*} FT \times T \xrightarrow{\pi_1} FT \tag{12}$$

Recall from Section 2.6 that τ^* is the extension of τ to a distributive law; intuitively, it is the inductive extension of τ to terms. Informally, the above natural transformation acts as follows. For an operator σ of arity n, given behaviour $k_1, \ldots, k_n \in FX \times X$ of its arguments, it first applies the assignment rule d to obtain a term $t(k_1, \ldots, k_n)$. Subsequently τ^* is used to compute the behaviour of t given the behaviour $k_1, \ldots, k_n$.

Definition 4.2. Given our fixed ρ and Δ, the map $\varphi\colon \mathbb{S} \to \mathbb{S}$ is defined as

$$\varphi(\tau) = \rho \vee \bigvee_{d \in \Delta} (\pi_1 \circ \tau^* \circ d_{F \times \mathsf{Id}}) \;:\; \Sigma(F \times \mathsf{Id}) \Rightarrow FT\,.$$

The reason for introducing φ is that we will compute its least fixed point, yielding a specification of interest. We first prove that φ is well-defined, and, to ensure the existence of a least fixed point, that it is monotone. The definition of φ should become more clear in Example 4.6, where we describe the construction for the concrete specification in Example 3.12.

For well-definedness, we need to check that φ preserves monotonicity. To this end, it is convenient to speak about monotonicity of a distributive law $\tau^*\colon T(F \times \mathsf{Id}) \Rightarrow (F \times \mathsf{Id})T$, which requires an order on T. Any partial order $(X, \leq)$ inductively extends to an order on TX by defining

$$\sigma(t_1, \ldots, t_n) \leq \tau(u_1, \ldots, u_m)$$

iff $\sigma = \tau$ (so also $n = m$) and $t_i \leq u_i$ for all $i \leq n$. We thus get a notion of monotonicity of distributive laws (this can be defined more generally using relation lifting, see [5]; here, we provide a concrete, self-contained exposition).

Lemma 4.3. *If τ is a monotone specification, then $\varphi(\tau)$ is monotone as well.*

Proof. We prove that if τ is monotone, then the distributive law $\tau^*\colon T(F \times \mathsf{Id}) \Rightarrow FT \times T$ is also monotone, by induction on pairs of terms $t, u \in T(FX \times X)$ with $t \leq u$ (note that this order is defined inductively). The desired result that $\varphi(\tau)$ is monotone then follows, since assignment rules d are clearly monotone.

For the base case, if $(b, x), (c, y) \in FX \times X$ with $(b, x) \leq (c, y)$ (so $b \leq c$ and $x = y$) then

$$\tau^*_X \circ \eta_{FX \times X}(b, x) = (F\eta_X \times \eta_X)(b, x) \leq (F\eta_X \times \eta_X)(c, y) = \tau^*_X \circ \eta_{FX \times X}(c, y)$$

where the inequality holds by monotonicity of $F\eta_X$ and since $x = y$, and the equalities by definition of τ^* (Equation (6) in Section 2.6).

Suppose σ is an operator of arity n, and we have terms $t_1, \ldots, t_n, u_1, \ldots, u_n \in T(FX \times X)$ with $\sigma(t_1, \ldots, t_n) \leq \sigma(u_1, \ldots, u_n)$, i.e., $t_i \leq u_i$ for all i. Further, suppose $\tau^*_X(t_i) \leq \tau^*_X(u_i)$ for all i. Then

$$\begin{array}{ll}
\tau^*_X \circ \nu_{FX \times X}(\sigma(t_1, \ldots, t_n)) & \\
= (F\mu_X \times \nu_X) \circ \langle \tau_{TX}, \Sigma\pi_2 \rangle \circ \Sigma\tau^*_X(\sigma(t_1, \ldots, t_n)) & \text{definition } \tau^* \\
= (F\mu_X \times \nu_X) \circ \langle \tau_{TX}, \Sigma\pi_2 \rangle(\sigma(\tau^*_X(t_1), \ldots, \tau^*_X(t_n))) & \text{definition } \Sigma \\
\leq (F\mu_X \times \nu_X) \circ \langle \tau_{TX}, \Sigma\pi_2 \rangle(\sigma(\tau^*_X(u_1), \ldots, \tau^*_X(u_n))) & \text{see below} \\
= \tau^*_X \circ \nu_{FX \times X}(\sigma(u_1, \ldots, u_n)) &
\end{array}$$

The inequality holds by monotonicity of $F\mu_X$ and τ, and the induction hypothesis; note that the induction hypothesis implies in particular $\pi_2 \circ \tau^*_X(t_i) = \pi_2 \circ \tau^*_X(u_i)$ for all i. □

Moreover, φ is monotone on $\mathbb{S}$:

Lemma 4.4. *The function $\varphi\colon \mathbb{S} \to \mathbb{S}$ is monotone.*

The main step in the proof of Lemma 4.4 is to show that the extension $(-)^*$ of abstract GSOS specifications to distributive laws is monotone.

Lemma 4.5. *Let τ_1, τ_2 be specifications. If $\tau_1 \leq \tau_2$ then $(\tau_1^*)_X \leq (\tau_2^*)_X$ for any set X.*

Proof. We have

$$(\tau_1^*)_X \circ \eta_{FX\times X} = F\eta_X \times \eta_X = (\tau_2^*)_X \circ \eta_{FX\times X}$$

by definition of $(-)^*$. Moreover

$$(F\mu_X \times \nu_X) \circ \langle (\tau_1)_{TX}, \Sigma\pi_2\rangle \leq (F\mu_X \times \nu_X) \circ \langle (\tau_2)_{TX}, \Sigma\pi_2\rangle$$

by monotonicity of $F\mu$ and assumption. Now using the definition of $(\tau_1^*)_X$, it easily follows by induction on terms in $T(FX \times X)$ that $(\tau_1^*)_X \leq (\tau_2^*)_X$. □

Because φ is monotone, it has a least fixed point, which we denote by $\mathsf{lfp}\,(\varphi)$.

Example 4.6. Consider the specification of the prefix, replication and parallel operator (and $\mathsf{0}$) in Example 3.12, based on a GSOS specification ρ and an assignment rule d. We compute the least fixed point of the associated function φ on specifications.

To gain some intuition, we first describe the least fixed point informally in terms of rules, and then compute it more precisely according to the definition of φ (Definition 4.2). In terms of rules, $\mathsf{lfp}\,(\varphi)$ will be the least GSOS specification such that:

1. it contains the rules from ρ, i.e., the two GSOS rules in (11);
2. if $!x|x \xrightarrow{a} t$ is derivable from some (positive) tests of variables H, then there is a rule
$$\frac{H}{!x \xrightarrow{a} t}$$
3. if $y|x \xrightarrow{a} t$ is derivable from some (positive) tests of variables H, then there is a rule
$$\frac{H}{x|y \xrightarrow{a} t}$$

The "least" way of satisfying the third item, is by having a rule

$$\frac{y \xrightarrow{a} y'}{x|y \xrightarrow{a} y'|x}$$

For the second item, given the behaviour of $x|y$ defined by the above rule and the one already in ρ, we can derive $!x|x \xrightarrow{a} t$ for some t only in two ways:

$$\frac{!x \xrightarrow{a} t}{!x|x \xrightarrow{a} t|x} \qquad \text{or} \qquad \frac{x \xrightarrow{a} x'}{!x|x \xrightarrow{a} x'|!x}$$

Hence we should have a rule as on the right-hand side below, and a rule as on the left-hand side whenever $!x \xrightarrow{a} t$ is derivable from some tests of variables H:

$$\frac{H}{!x \xrightarrow{a} t|x} \qquad \frac{x \xrightarrow{a} x'}{!x \xrightarrow{a} x'|!x}$$

The least specification satisfying this has (in addition to the above rule on the right-hand side) a rule

$$\frac{x \xrightarrow{a} x'}{!x \xrightarrow{a} x'|!x|x^i}$$

for each i, where x^i is the i-fold parallel composition; strictly seen this is left associative, i.e., the expression is of the form $(((x'|!x)|x)|\ldots|x)$.

Next, we describe the above construction more precisely for ρ and d, by computing $\mathsf{lfp}\,(\varphi)$. For any $(f,x),(g,y) \in FX \times X$:

$$\begin{array}{lll} \pi_1 \circ \tau_X^* \circ d_{FX\times X}((f,x)|(g,y)) & = \pi_1 \circ \tau_X^*((g,y)|(f,x)) & \text{def. } d \\ & = \tau_X((g,y)|(f,x)) & (7), \text{ Sect. 2.6} \end{array}$$

Hence, if $\varphi(\tau) = \tau$ then

$$\tau_X((f,x)|(g,y)) = \rho_X((f,x)|(g,y)) \vee \tau_X((g,y)|(f,x))$$

and it follows that

$$\mathsf{lfp}\,(\varphi)_X((f,x)|(g,y)) = \rho_X((f,x)|(g,y)) \vee \rho_X((g,y)|(f,x))\,. \tag{13}$$

Also, it is easy to compute that $(\mathsf{lfp}\,(\varphi))_X(\mathsf{0}) = \rho_X(\mathsf{0})$ and $(\mathsf{lfp}\,(\varphi))_X(a.(f,x)) = \rho_X(a.(f,x))$. Now, for the replication operator:

$$\begin{array}{l} \pi_1 \circ (\mathsf{lfp}\,(\varphi))_X^* \circ d_{FX\times X}(!(f,x)) \\ = \pi_1 \circ (\mathsf{lfp}\,(\varphi))_X^*(!(f,x)|(f,x)) \\ = F\mu_X \circ (\mathsf{lfp}\,(\varphi))_{TX} \circ \Sigma(\mathsf{lfp}\,(\varphi))_X^*(!(f,x)|(f,x)) \\ = F\mu_X \circ (\mathsf{lfp}\,(\varphi))_{TX}((\mathsf{lfp}\,(\varphi))_X^*(!(f,x))|(\mathsf{lfp}\,(\varphi))_X^*(f,x)) \\ = F\mu_X \circ (\mathsf{lfp}\,(\varphi))_{TX}(((\mathsf{lfp}\,(\varphi))_X(!(f,x)),!x)|((F\eta_X)(f),\eta_X(x))) \\ = F\mu_X \circ (\rho_X(((\mathsf{lfp}\,(\varphi))_X(!(f,x)),!x)|((F\eta_X)(f),\eta_X(x))) \\ \qquad\qquad \vee \rho_X(((F\eta_X)(f),\eta_X(x))|((\mathsf{lfp}\,(\varphi))_X(!(f,x)),!x))) \\ = F\mu_X(\lambda a.\{x'|\eta_X(x) \mid x' \in (\mathsf{lfp}\,(\varphi))_X(!(f,x))(a)\} \cup \{x'|!x \mid x' \in (F\eta_X)(f)(a)\}) \\ = \lambda a.\{x'|x \mid x' \in (\mathsf{lfp}\,(\varphi))_X(!(f,x))(a)\} \cup \{x'|!x \mid x' \in f(a)\} \end{array}$$

where the first equality is by definition of d, the second by (6) in Section 2.6, the third by definition of Σ, the fourth by (7) (the left part) and (6) (the right part), the fifth by (13), the sixth by definition of ρ, and the final equality by definition and laws of η and μ.

By the above computation and the fact that $\rho_X(!(f,x)) = \lambda a.\emptyset$ (see Example 3.12), $(\mathsf{lfp}\,(\varphi))_X(!(f,x))$ is the least function such that for all $a \in A$:

$$\{x'|x \mid x' \in (\mathsf{lfp}\,(\varphi))_X(!(f,x))(a)\} \cup \{x'|!x \mid x' \in f(a)\} \subseteq (\mathsf{lfp}\,(\varphi))_X(!(f,x))(a)$$

which is given by $(\mathsf{lfp}\,(\varphi))_X(!(f,x)) = \{x' | !x | x^i \mid i \in \mathbb{N}\}$, where x^i is the i-fold parallel composition (with brackets as explained above). This corresponds to the concrete set of rules that we have seen in the informal explanation in the first half of this example.

Since φ preserves monotonicity we obtain monotonicity of $\mathsf{lfp}\,(\varphi)$ by transfinite induction (the base case and limit steps are rather easy). Here we use the fact stated in Section 2.1 that least fixed point of a monotone function in a complete lattice can be constructed as the supremum of an ascending chain obtained by iterating the function over the ordinals.

Corollary 4.7. *The abstract GSOS specification* $\mathsf{lfp}\,(\varphi)$ *is monotone.*

We proceed to prove that the operational model of the least fixed point of φ is precisely the interpretation of ρ and Δ (the least fixed point of ψ as given in Definition 3.9), i.e., that $M(\mathsf{lfp}\,(\varphi)) = \mathsf{lfp}\,(\psi)$. First, we show that $M(\mathsf{lfp}\,(\varphi))$ is a fixed point of ψ.

Lemma 4.8. *The operational model* $M(\mathsf{lfp}\,(\varphi))$ *of the specification* $\mathsf{lfp}\,(\varphi)$ *is a* (ρ, Δ)*-model, i.e.,* $\psi(M(\mathsf{lfp}\,(\varphi))) = M(\mathsf{lfp}\,(\varphi))$.

Proof. Let $f = M(\mathsf{lfp}\,(\varphi))$. We must show that $\psi(f) = f$.

$$
\begin{aligned}
& f \circ \nu_\emptyset \\
&= F\mu_\emptyset \circ (\mathsf{lfp}\,(\varphi))_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \\
&= F\mu_\emptyset \circ (\rho \vee \textstyle\bigvee_{d\in\Delta} \pi_1 \circ (\mathsf{lfp}\,(\varphi))^* \circ d_{F\times \mathsf{Id}})_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \\
&= F\mu_\emptyset \circ (\rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \textstyle\bigvee_{d\in\Delta} \pi_1 \circ (\mathsf{lfp}\,(\varphi))^*_{T\emptyset} \circ d_{FT\emptyset\times T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle) \\
&= F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \textstyle\bigvee_{d\in\Delta} F\mu_\emptyset \circ \pi_1 \circ (\mathsf{lfp}\,(\varphi))^*_{T\emptyset} \circ d_{FT\emptyset\times T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle
\end{aligned}
$$

where the first equality holds by definition of M, the second since $\mathsf{lfp}\,(\varphi)$ is a fixed point of φ, the third holds by the definition of the join on natural transformations and the last one holds by the fact the $F\mu_\emptyset$ preserves joins. For the right-hand part, we have

$$
\begin{aligned}
& \textstyle\bigvee_{d\in\Delta} F\mu_\emptyset \circ \pi_1 \circ (\mathsf{lfp}\,(\varphi))^*_{T\emptyset} \circ d_{FT\emptyset\times T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \\
&= \textstyle\bigvee_{d\in\Delta} \pi_1 \circ F\mu_\emptyset \times \mu_\emptyset \circ (\mathsf{lfp}\,(\varphi))^*_{T\emptyset} \circ T\langle f, \mathsf{id}\rangle \circ d_{T\emptyset} && \text{naturality of } d,\ \pi_1 \\
&= \textstyle\bigvee_{d\in\Delta} \pi_1 \circ \langle f, \mathsf{id}\rangle \circ \mu_\emptyset \circ d_{T\emptyset} && \text{equation (8)} \\
&= \textstyle\bigvee_{d\in\Delta} f \circ \mu_\emptyset \circ d_{T\emptyset}
\end{aligned}
$$

Thus $f\circ\nu_\emptyset = F\mu_\emptyset\circ\rho_{T\emptyset}\circ\Sigma\langle f, \mathsf{id}\rangle\vee\bigvee_{d\in\Delta} f\circ\mu_\emptyset\circ d_{T\emptyset} = \psi(f)\circ\nu_\emptyset$ and consequently $\psi(f) = f$, since $\nu_\emptyset$ is an isomorphism. □

We proceed to show that $M(\mathsf{lfp}\,(\varphi)) \leq \mathsf{lfp}\,(\psi)$. Since $\psi(M(\mathsf{lfp}\,(\varphi))) = M(\mathsf{lfp}\,(\varphi))$ by the above Lemma 4.8, we then have $M(\mathsf{lfp}\,(\varphi)) = \mathsf{lfp}\,(\psi)$ (Theorem 4.14). The main step is that any fixed point of ψ is "closed under ρ", i.e., that in such a model, all behaviour that we can derive by the specification is already there. This result is the contents of Corollary 4.13 below; it follows by transfinite induction from Lemma 4.11 and 4.12. But first, we need a few technical tools. Recall from Section 2.2 that a Σ-algebra $\alpha\colon \Sigma X \to X$ induces an algebra $\widehat{\alpha}\colon TX \to X$. This construction preserves algebra morphisms. We prove a lax version of this fact.

Lemma 4.9. *Let $\alpha\colon \Sigma X \to X$ and $\beta\colon \Sigma Y \to Y$ be algebras, such that Y carries a partial order $\leq$ and β is monotone. Then for any function $f\colon X \to Y$:*

$$\begin{array}{ccc} \Sigma X & \xrightarrow{\Sigma f} & \Sigma Y \\ {\scriptstyle\alpha}\downarrow & \geq & \downarrow{\scriptstyle\beta} \\ X & \xrightarrow[f]{} & Y \end{array} \qquad \textit{implies} \qquad \begin{array}{ccc} TX & \xrightarrow{Tf} & TY \\ {\scriptstyle\widehat{\alpha}}\downarrow & \geq & \downarrow{\scriptstyle\widehat{\beta}} \\ X & \xrightarrow[f]{} & Y \end{array}$$

where the diagrams denote pointwise inequality, i.e., the left-hand side means that for all $t \in \Sigma X$: $f \circ \alpha(t) \geq \beta \circ \Sigma f(t)$, and similarly for the right-hand side.

Proof. Suppose $\beta \circ \Sigma f \leq f \circ \alpha$. The proof is by induction on $t \in TX$. For the base case $t = \eta_X(s) \in TX$, we have an equality, without using the assumption:

$$\widehat{\beta} \circ Tf \circ \eta_X(s) = \widehat{\beta} \circ \eta_Y \circ f(s) = f(s) = f \circ \widehat{\alpha} \circ \eta_X(s)\,.$$

Now suppose $\sigma \in \Sigma$ is of arity n, and for some $t_1, \ldots, t_n \in TX$, we have $\widehat{\beta} \circ (Tf)(t_i) \leq f \circ \widehat{\alpha}(t_i)$ for all i with $1 \leq i \leq n$. Then

$$\begin{array}{ll} \widehat{\beta} \circ Tf \circ \nu_X(\sigma(t_1, \ldots, t_n)) & \\ = \widehat{\beta} \circ \nu_Y \circ \Sigma Tf(\sigma(t_1, \ldots, t_n)) & \text{naturality } \nu \\ = \widehat{\beta} \circ \nu_Y(\sigma(Tf(t_1), \ldots, Tf(t_n))) & \text{definition } \Sigma \\ = \beta \circ \Sigma\widehat{\beta}(\sigma(Tf(t_1), \ldots, Tf(t_n))) & \text{definition } \widehat{\beta} \\ = \beta(\sigma(\widehat{\beta} \circ Tf(t_1), \ldots, \widehat{\beta} \circ Tf(t_n))) & \text{definition } \Sigma \\ \leq \beta(\sigma(f \circ \widehat{\alpha}(t_1), \ldots, f \circ \widehat{\alpha}(t_n))) & \text{ind. hypothesis, monotonicity } \beta \\ = \beta \circ \Sigma f \circ \Sigma\widehat{\alpha}(\sigma(t_1, \ldots, t_n)) & \text{definition } \Sigma \\ \leq f \circ \alpha \circ \Sigma\widehat{\alpha}(\sigma(t_1, \ldots, t_n)) & \text{assumption} \\ = f \circ \widehat{\alpha} \circ \nu_X(\sigma(t_1, \ldots, t_n)) & \text{definition } \widehat{\alpha} \end{array}$$

which concludes the induction step. □

We instantiate the above lemma to the definition of τ^*.

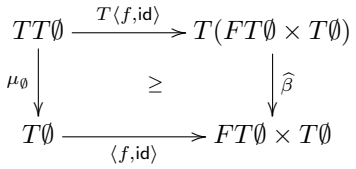
Lemma 4.10. *Let τ be a monotone abstract GSOS specification of Σ over F. Then for any $f\colon T\emptyset \to FT\emptyset$:*

$$\begin{array}{ccc} \Sigma T\emptyset & \xrightarrow{\Sigma\langle f, \mathsf{id}\rangle} & \Sigma(FT\emptyset \times T\emptyset) \\ & & \downarrow{\scriptstyle\tau_{T\emptyset}} \\ {\scriptstyle\nu_\emptyset}\downarrow & \geq & FTT\emptyset \\ & & \downarrow{\scriptstyle F\mu_\emptyset} \\ T\emptyset & \xrightarrow[f]{} & FT\emptyset \end{array} \qquad \textit{implies} \qquad \begin{array}{ccc} TT\emptyset & \xrightarrow{T\langle f, \mathsf{id}\rangle} & T(FT\emptyset \times T\emptyset) \\ & & \downarrow{\scriptstyle\tau^*_{T\emptyset}} \\ {\scriptstyle\mu_\emptyset}\downarrow & \geq & FTT\emptyset \times TT\emptyset \\ & & \downarrow{\scriptstyle F\mu_\emptyset \times \mu_\emptyset} \\ T\emptyset & \xrightarrow[\langle f, \mathsf{id}\rangle]{} & FT\emptyset \times T\emptyset \end{array}$$

Proof. From the assumption it follows that

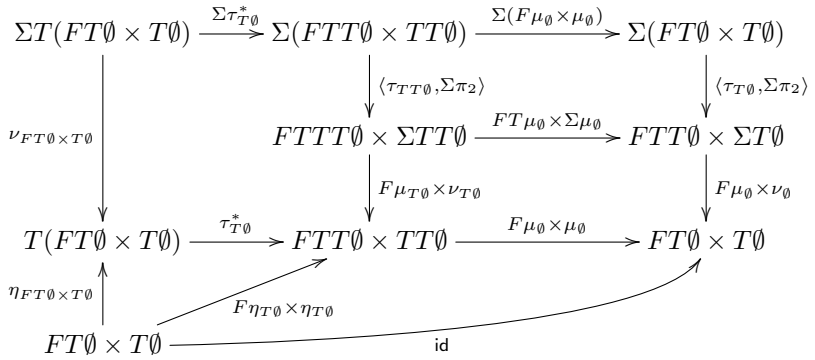
$$(F\mu_\emptyset \times \nu_\emptyset) \circ \langle \tau_{T\emptyset}, \Sigma\pi_2\rangle \circ \Sigma\langle f, \mathsf{id}\rangle \leq \langle f, \mathsf{id}\rangle \circ \nu_\emptyset\,.$$

Let $\beta = (F\mu_\emptyset \times \nu_\emptyset) \circ \langle \tau_{T\emptyset}, \Sigma\pi_2 \rangle$, then by Lemma 4.9 we get



where $\widehat{\beta}$ is the T-algebra induced by the Σ-algebra $\beta = (F\mu_\emptyset \times \nu_\emptyset) \circ \langle \tau_{T\emptyset}, \Sigma\pi_2 \rangle$. Thus, it only remains to prove that $\widehat{\beta} = (F\mu_\emptyset \times \mu_\emptyset) \circ \tau^*_{T\emptyset}$.

To this end, consider the following diagram:



The upper right rectangle commutes by naturality, the lower right rectangle commutes by the multiplication law of the monad and since $\mu_\emptyset = \widehat{\nu_\emptyset}$ (see Section 2.2). The left square and lower left triangle commute by definition of τ^* (Equation (6) in Section 2.6), and the lower right triangle by a unit law of the monad. Thus $(F\mu_\emptyset \times \mu_\emptyset) \circ \tau^*_{T\emptyset}$ is an algebra homomorphism extending id, and since $\widehat{\beta}$ is by definition an algebra homomorphism extending id and homomorphic extensions are unique, we have $\widehat{\beta} = (F\mu_\emptyset \times \mu_\emptyset) \circ \tau^*_{T\emptyset}$. □

Lemma 4.11. *Let τ be a specification, and $f \in \mathbb{M}$ a fixed point of ψ. If $F\mu_\emptyset \circ \tau_{T\emptyset} \circ \Sigma\langle f, \mathsf{id} \rangle \leq f \circ \nu_\emptyset$ then $F\mu_\emptyset \circ \varphi(\tau)_{T\emptyset} \circ \Sigma\langle f, \mathsf{id} \rangle \leq f \circ \nu_\emptyset$.*

Proof.

$$
\begin{aligned}
&F\mu_\emptyset \circ \varphi(\tau)_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \\
&= F\mu_\emptyset \circ (\rho \vee \bigvee_{d\in\Delta} \pi_1 \circ \tau^* \circ d_{F\times \mathsf{Id}})_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \\
&= F\mu_\emptyset \circ (\rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{d\in\Delta} \pi_1 \circ \tau^*_{T\emptyset} \circ d_{FT\emptyset\times T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle) \\
&= F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{d\in\Delta} F\mu_\emptyset \circ \pi_1 \circ \tau^*_{T\emptyset} \circ d_{FT\emptyset\times T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \\
&= F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{d\in\Delta} \pi_1 \circ (F\mu_\emptyset \times \mu_\emptyset) \circ \tau^*_{T\emptyset} \circ T\langle f, \mathsf{id}\rangle \circ d_{T\emptyset} \\
&\leq F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{d\in\Delta} \pi_1 \circ \langle f, \mathsf{id}\rangle \circ \mu_\emptyset \circ d_{T\emptyset} \\
&= F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{d\in\Delta} f \circ \mu_\emptyset \circ d_{T\emptyset} \\
&= \psi(f) \circ \nu_\emptyset = f \circ \nu_\emptyset
\end{aligned}
$$

The first equality holds by definition of φ, the second by definition of the join of specifications, the third since $F\mu_\emptyset$ is join-preserving, and the fourth equality by naturality of d and π_1. The inequality holds by assumption and Lemma 4.10. The last equality holds by definition of ψ. □

Lemma 4.12. *Let $f \in \mathbb{M}$ be a fixed point of ψ, and suppose we have a family $\{\tau_i\}_{i\in I}$ of specifications, for some index set I. If $F\mu_\emptyset \circ (\tau_i)_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq f \circ \nu_\emptyset$ for all $i \in I$, then $F\mu_\emptyset \circ (\bigvee_{i\in I} \tau_i)_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq f \circ \nu_\emptyset$.*

Proof. Since $F\mu_\emptyset$ preserves joins we have

$$
F\mu_\emptyset \circ (\bigvee_{i\in I} \tau_i)_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle = \bigvee_{i\in I} F\mu_\emptyset \circ (\tau_i)_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle
$$

and the result now follows by the assumption that $F\mu_\emptyset \circ (\tau_i)_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq f \circ \nu_\emptyset$ for each i. □

Corollary 4.13. *If $f \in \mathbb{M}$ is a fixed point of ψ, then:*

$$
\begin{array}{ccc}
\Sigma T\emptyset & \xrightarrow{\Sigma\langle f, \mathsf{id}\rangle} & \Sigma(FT\emptyset \times T\emptyset) \\
\Big\downarrow{\scriptstyle \nu_\emptyset} & \geq & \Big\downarrow{\scriptstyle \mathsf{lfp}\,(\varphi)_{T\emptyset}} \\
T\emptyset & & FTT\emptyset \\
 & & \Big\downarrow{\scriptstyle F\mu_\emptyset} \\
T\emptyset & \xrightarrow[f]{} & FT\emptyset
\end{array}
$$

Proof. By transfinite induction. For the base case we have $F\mu_\emptyset \circ \bot \circ \Sigma\langle f, \mathsf{id}\rangle = \bot \leq f \circ \nu_\emptyset$. The successor step is given by Lemma 4.11 and the limit step by Lemma 4.12. □

This allows to prove our main result.

Theorem 4.14. *The interpretation of ρ and Δ coincides with the operational model of the specification $\mathsf{lfp}\,(\varphi)$, i.e., $M(\mathsf{lfp}\,(\varphi)) = \mathsf{lfp}\,(\psi)$.*

Proof. By Lemma 4.8, $M(\mathsf{lfp}\,(\varphi))$ is a fixed point of ψ. To show it is the least one, let f be any fixed point of ψ; we proceed to prove $M(\mathsf{lfp}\,(\varphi)) \leq f$ by structural induction on closed terms. Suppose $\sigma \in \Sigma$ is an operator of arity n, and suppose we have $t_1, \ldots, t_n \in T\emptyset$ such that $M(\mathsf{lfp}\,(\varphi))(t_i) \leq f(t_i)$ for all i with $1 \leq i \leq n$ (note that this trivially holds in the base case, when $n = 0$).

Since $M(\mathsf{lfp}\,(\varphi))$ is the operational model of $\mathsf{lfp}\,(\varphi)$, we have

$$\begin{aligned} &M(\mathsf{lfp}\,(\varphi))(\sigma(t_1, \ldots, t_n)) \\ &= F\mu_\emptyset \circ (\mathsf{lfp}\,(\varphi))_{T\emptyset} \circ \Sigma\langle M(\mathsf{lfp}\,(\varphi)), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n))\,. \end{aligned} \tag{14}$$

From the induction hypothesis we have

$$\Sigma\langle M(\mathsf{lfp}\,(\varphi)), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) \leq \Sigma\langle f, \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n))\,.$$

By monotonicity of $F\mu_\emptyset$ and $\mathsf{lfp}\,(\varphi)$ (Corollary 4.7) we then obtain

$$\begin{aligned} &F\mu_\emptyset \circ (\mathsf{lfp}\,(\varphi))_{T\emptyset} \circ \Sigma\langle M(\mathsf{lfp}\,(\varphi)), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) \\ &\leq F\mu_\emptyset \circ (\mathsf{lfp}\,(\varphi))_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n))\,. \end{aligned} \tag{15}$$

By Corollary 4.13, we have

$$F\mu_\emptyset \circ (\mathsf{lfp}\,(\varphi))_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) \leq f(\sigma(t_1, \ldots, t_n))\,. \tag{16}$$

Combining (14), (15) and (16) we obtain

$$M(\mathsf{lfp}\,(\varphi))(\sigma(t_1, \ldots, t_n)) \leq f(\sigma(t_1, \ldots, t_n))$$

as desired. □

As a consequence, the interpretation of ρ and Δ is well-behaved.

Corollary 4.15. *Bisimilarity is a congruence on the interpretation $\mathsf{lfp}\,(\psi)$ of ρ and Δ, and bisimulation up to context is compatible (more precisely, the contextual closure is $b_{\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle}$-compatible).*

Example 4.16. In Example 3.12, we have seen the specification of a basic process calculus with parallel composition and replication, in terms of GSOS rules and assignments, and it was shown that its desired interpretation is given by $\mathsf{lfp}\,(\psi)$. By Theorem 4.14, this interpretation is the operational model of the constructed GSOS specification $\mathsf{lfp}\,(\varphi)$. Hence, bisimilarity is a congruence, and the contextual closure is $b_{\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle}$-compatible.

It is easy to extend Example 3.12 to include the usual parallel operator from CCS (with synchronization, which we omitted to simplify the presentation a bit), and to add rules for restriction and choice. By the above Corollary, we obtain once more that bisimilarity is a congruence and the contextual closure is compatible, for CCS with replication. This compatibility result is known (see, e.g., [32]), but contrary to previous proofs, here we obtain it directly from the fact that the specification is expressed in terms of GSOS and assignment rules, by the above results.

Example 4.17. We consider a fragment of the "general process algebra with transitions costs" (GPA) from [7]. The set P of basic GPA processes is defined by the grammar

$$t ::= 0 \mid t + t \mid (a, r).t \mid p$$

where a ranges over the set of actions A, r ranges over the positive real numbers $\mathbb{R}^+$, and p ranges over a fixed set of procedure names $PNames$. We assume that each procedure name $p_i \in PNames$ has a body $t_i \in P$.

The operational semantics of basic GPA processes on the monoid $\mathbb{R}^+ \cup \{\infty\}$ (with supremum) is given by the coalgebra $\alpha \colon P \to ((\mathbb{R}^+ \cup \{\infty\})^P)^A$, defined for all $a' \in A$ and $t' \in P$ as follows:

$$\begin{aligned} \alpha(0)(a')(t') &= 0 \\ \alpha((a,r).t)(a')(t') &= \begin{cases} r & \text{if } a = a', t = t' \\ 0 & \text{otherwise} \end{cases} \\ \alpha(t_1 + t_2)(a')(t') &= \mathsf{sup}\{\alpha(t_1)(a')(t'), \alpha(t_2)(a')(t')\} \end{aligned}$$

Equivalently, it is described by the following rules:

$$\frac{r \neq 0}{(a,r).p \xrightarrow{a,r} p} \qquad \frac{p_1 \xrightarrow{a,r} p' \quad r \neq 0}{p_1 + p_2 \xrightarrow{a,s} p'} \qquad \frac{p_2 \xrightarrow{a,r} p' \quad r \neq 0}{p_1 + p_2 \xrightarrow{a,s} p'}$$

where $s = \mathsf{sup}\{\alpha(p_1)(a)(p'), \alpha(p_2)(a)(p')\}$. This defines an abstract GSOS specification, which is monotone, with α as its unique operational model. The (recursive) procedures can now be interpreted by assignment rules, i.e., for each $p_i \in PNames$ we add an assignment rule $p_i := t_i$. Intuitively this means that the procedure call p_i is given by the behaviour of its body t_i, as expected. By Theorem 4.14, bisimilarity is a congruence on α. Note that we can replace one of the two rules for $+$ by the assignment rule $x + y := y + x$.

5. Structural Congruences (as Assignment Rules)

The assignment rules considered in the theory of the previous sections copy behaviour from a term to an operator, but this assignment goes one way only. In this section, we consider the combination of abstract GSOS specifications with actual *equations*, interpreted by the structural congruence rule. By encoding equations in a restricted format as assignment rules, we obtain that the interpretation of any specification with equations in this format is well-behaved.

Equations are elements of $TV \times TV$, where V is an arbitrary but fixed set of variables. A set of equations $E \subseteq TV \times TV$ induces a *congruence* $\equiv_E$:

Definition 5.1. Let $E \subseteq TV \times TV$ be a set of equations. The *congruence closure* of E is the least relation $\equiv_E \subseteq T\emptyset \times T\emptyset$ satisfying the following rules:

$$\frac{t\,E\,u \quad s\colon V \to T\emptyset}{s^\sharp(t) \equiv_E s^\sharp(u)} \qquad \frac{}{t \equiv_E t} \qquad \frac{u \equiv_E t}{t \equiv_E u} \qquad \frac{t \equiv_E u \qquad u \equiv_E v}{t \equiv_E v}$$

$$\frac{t_1 \equiv_E u_1 \qquad \dots \qquad t_n \equiv_E u_n}{\sigma(t_1, \dots, t_n) \equiv_E \sigma(u_1, \dots, u_n)} \qquad \text{for each } \sigma \in \Sigma, n = |\sigma|$$

where $s^\sharp$ is the extension of s to terms (Section 2.2).

In the context of structural operational semantics, equations are often interpreted by the *structural congruence rule*:

$$\frac{t \equiv_E u \qquad u \xrightarrow{a} u' \qquad u' \equiv_E v}{t \xrightarrow{a} v} \tag{17}$$

Informally, this rule states that we can deduce transitions modulo the congruence generated by the equations. In fact, replacing this rule by

$$\frac{t \equiv_E u \qquad u \xrightarrow{a} u'}{t \xrightarrow{a} u'} \tag{18}$$

does not affect the behaviour, modulo bisimilarity [30]. See loc. cit. for details on the interpretation of structural congruences in the context of transition systems.

We denote by $(T\emptyset)/\equiv_E$ the set of equivalence classes, and by $q\colon T\emptyset \to (T\emptyset)/\equiv_E$ the quotient map of $\equiv_E$ (we remark that one can equip $(T\emptyset)/\equiv_E$ with an algebra structure μ' such that q is a T-algebra homomorphism). Thus $q(t) = q(u)$ iff $t \equiv_E u$. Further $t \equiv_E u$ iff there is a right inverse $r\colon (T\emptyset)/\equiv_E \to T\emptyset$ of q such that $r(q(t)) = u$. The latter fact is exploited in the interpretation of a specification together with a set of equations.

Definition 5.2. Let $\theta\colon \mathbb{M} \to \mathbb{M}$ be the (unique) function such that

$$\theta(f) \circ \nu_\emptyset = F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{r \in R} f \circ r \circ q \circ \nu_\emptyset \;:\; \Sigma T\emptyset \to FT\emptyset$$

where R is the set of right inverses of q. A (ρ, E)-model is a coalgebra $f \in \mathbb{M}$ such that $\theta(f) = f$.

Lemma 5.3. *The function θ is monotone.*

Proof. Similar to the proof of Lemma 3.10. □

Definition 5.4. The *interpretation* of ρ and E is the least (ρ, E)-model.

The interpretation is also the least pre-fixed point of θ, i.e., the least coalgebra $f\colon T\emptyset \to FT\emptyset$ such that $\theta(f) \leq f$. Notice that, by the discussion above Definition 5.2, we have $\bigvee_{r\in R} f \circ r \circ q \circ \nu_\emptyset \leq f \circ \nu_\emptyset$ if and only if for all $t, u \in T\emptyset$: $t \equiv_E u$ implies $f(u) \leq f(t)$. Since $\equiv_E$ is symmetric and $\leq$ is anti-symmetric, this is again equivalent to the property that for all $t, u \in T\emptyset$: $t \equiv_E u$ implies $f(t) = f(u)$. The interpretation is thus the least coalgebra $f\colon T\emptyset \to FT\emptyset$ such that

1. $F\mu_\emptyset \circ \rho_{T\emptyset}(\sigma((f(t_1), t_1), \ldots, (f(t_n), t_n))) \leq f(\sigma(t_1, \ldots, t_n))$ for every operator σ (with arity n) and $t_1, \ldots, t_n \in T\emptyset$;
2. if $t \equiv_E u$ then $f(t) = f(u)$.

The second part corresponds to the intuitive interpretation of the rule (18).

Example 5.5. Consider again the process calculus with operators $\mathsf{0}$, $a.x$, $x|y$ and $!x$. We specify the semantics by GSOS rules and equations:

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{x|y \xrightarrow{a} x'|y} \qquad x|y = y|x \qquad !x = !x|x \tag{19}$$

This is minor variation on the specification in Example 3.12, obtained by replacing assignment rules by equations. It is explained in Example 3.12 how the GSOS rules give rise to a natural transformation ρ. Using the characterization of the interpretation $f\colon T\emptyset \to FT\emptyset$ of ρ with equations described in the text above the current example, we have that f is the least transition system such that:

1. for all $a \in A$ and $t, u \in T\emptyset$:

$$t \in f(a.t)(a) \qquad \text{and} \qquad \{t'|u \mid t' \in f(t)(a)\} \subseteq f(t|u)(a)$$

   (this was already computed in Example 3.12), and
2. if $t \equiv_E u$ then $f(t) = f(u)$, where $\equiv_E$ is the congruence generated by the equations $x|y = y|x$ and $!x = !x|x$.

This is the desired interpretation of (19), where the equations are interpreted according to (18).

In general, bisimilarity is not a congruence when equations are added. For convenience we recall a counterexample on transition systems [30].

Example 5.6. Consider rules

$$\frac{}{p \xrightarrow{a} p} \quad \text{and} \quad \frac{}{q \xrightarrow{a} p}$$

and the single equation $p = \sigma(q)$, where p, q are constants, σ is a unary operator and a is an arbitrary label. In the interpretation, p is bisimilar to q, but $\sigma(p)$ is not bisimilar to $\sigma(q)$.

The above counterexample is based on assigning behaviour to the term $\sigma(q)$, rather than defining each operator independently of the syntax of its arguments. To rule out such assignments, a restricted format of equations was introduced in [30], called cfsc. The main result of [30] is that for any specification in the tyft format combined with cfsc equations, bisimilarity is a congruence.

Definition 5.7. A set of equations $E \subseteq TV \times TV$ is in the cfsc format with respect to ρ if every equation is of one of the following forms:

1. A *σx-equation*: $\sigma_1(x_1, \ldots, x_n) = \sigma_2(y_1, \ldots, y_n)$, where $\sigma_1, \sigma_2 \in \Sigma$ are of arity n (possibly $\sigma_1 = \sigma_2$), $x_1, \ldots, x_n$ are distinct variables and $y_1, \ldots, y_n$ is a permutation of $x_1, \ldots, x_n$.
2. A *defining equation*: $\sigma(x_1, \ldots, x_n) = t$ where $\sigma \in \Sigma$ and t is an arbitrary term (which may involve σ again); $x_1, \ldots, x_n$ are distinct variables, and all variables that occur in t are in $x_1, \ldots, x_n$. Moreover σ does not appear in any other equation in E, and $\rho_X(\sigma(u_1, \ldots, u_n)) = \bot$ for any set X and any $u_1, \ldots, u_n \in FX \times X$.

A *σx-equation* allows to assign simple algebraic properties to operators which already have behaviour; the prototypical example here is commutativity, like in the specification of the parallel composition in (2). With a *defining equation*, as the name suggests, one can define the behaviour of an operator. An example is $!x = !x|x$; another example is $p = q|z|a.p$ where p, q and z are constants. Further, the procedure declarations of Example 4.17 can be modelled by defining equations. Associativity of $|$ is neither a σx-equation nor a defining one. We refer to [30] for arguments that the cfsc format cannot be trivially extended. The cfsc format depends on an abstract GSOS specification: operators at the left hand side of a defining equation should not get any behaviour in the specification. This restriction ensures that one cannot assign behaviour to complex terms, disallowing a situation such as in Example 5.6.

We proceed to show that the interpretation of an abstract GSOS specification ρ and a set of equations E in the cfsc format equals the operational model of a certain other specification, up to bisimilarity. This is done by encoding equations in this format as assignment rules, and using the theory of the previous section to obtain the desired result.

First, note that for any σx-equation $\sigma_1(x_1, \ldots, x_n) = \sigma_2(y_1, \ldots, y_n)$, the variables on one side are a permutation of the variables on the other, hence a σx-equation can equivalently be represented as a triple (σ_1, σ_2, p) where $p\colon \mathsf{Id}^n \to \mathsf{Id}^n$ is the natural transformation corresponding to the permutation of variables in the equation. where $p\colon \mathsf{Id}^n \to \mathsf{Id}^n$ is the natural transformation corresponding to the permutation given by the equation. Below, we use $t[x_1, \ldots, x_n := t_1, \ldots, t_n]$ to denote the simultaneous substitution of variables $x_1, \ldots, x_n$ by terms $t_1, \ldots, t_n$ in a term t.

Construction 5.8. A set of equations E in the cfsc format defines a set of assignment rules Δ^E as follows:

1. For every σx-equation (σ_1, σ_2, p) we define d and d' on a component X as

$$d_X(\sigma(u_1, \ldots, u_n)) = \begin{cases} \sigma_2(p_X(u_1, \ldots, u_n)) & \text{if } \sigma = \sigma_1 \\ \sigma(u_1, \ldots, u_n) & \text{otherwise} \end{cases}$$

   for all $u_1, \ldots, u_n \in X$, and d' is similarly defined using the inverse permutation p^{-1}, with σ_1 and σ_2 swapped.
2. For every defining equation $\sigma_1(x_1, \ldots, x_n) = t$ we define a corresponding assignment rule

$$d_X(\sigma(u_1, \ldots, u_n)) = \begin{cases} t[x_1, \ldots, x_n := u_1, \ldots, u_n] & \text{if } \sigma = \sigma_1 \\ \sigma(u_1, \ldots, u_n) & \text{otherwise} \end{cases}$$

   for any set X and all $u_1, \ldots, u_n \in X$.

Remark 5.9. In [30], σx-equations are a bit more liberal in that they do not require the arities of σ and σ' to coincide, and do allow variables which only occur on one side of the equation. But in the interpretation these variables are quantified universally over closed terms; thus, we could encode this using infinitely many assignment rules. For example, an equation $\sigma_1(x) = \sigma_2(x, y)$ can be encoded by the set of assignment rules, one for each term $t \in T\emptyset$, mapping $\sigma_1(x)$ to $\sigma_2(x, t)$ (and the single assignment rule mapping $\sigma_2(x, y)$ to $\sigma_1(x)$). We work with the simpler format above for technical convenience.

We prove that the encoding of equations as assignment rules in Construction 5.8 is correct with respect to the interpretation of the equations (Theorem 5.13). First, we show that if $\sigma(x_1, \ldots, x_n) = t$ is a defining equation of a set of equations in the cfsc format, then the behaviour of $\sigma(x_1, \ldots, x_n)$ will be below that of t.

Lemma 5.10. *Let E be a set of equations in the cfsc format w.r.t. ρ, and let ψ be as in Definition 3.9 for (ρ, Δ^E). Then for any defining equation $\sigma(x_1, \ldots, x_n) = t$ and any $t_1, \ldots, t_n \in T\emptyset$: $\mathsf{lfp}(\psi) \circ \nu_\emptyset(\sigma(t_1, \ldots, t_n)) \leq \mathsf{lfp}(\psi) \circ \mu_\emptyset(t[x_1 := t_1, \ldots, x_n := t_n])$.*

Proof. Given a defining equation, let $d \in \Delta^E$ be the natural transformation that encodes it (see Construction 5.8(2)). We prove by transfinite induction that for any function $g \in \mathbb{M}$ that arises in the iterative construction of $\mathsf{lfp}(\psi)$ and for any $t_1, \ldots, t_n \in T\emptyset$ we have

$$g \circ \nu_\emptyset(\sigma(t_1, \ldots, t_n)) \leq \mathsf{lfp}(\psi) \circ \mu_\emptyset \circ d_{T\emptyset}(\sigma(t_1, \ldots, t_n)) \,. \tag{20}$$

The base case is when $g = \bot$, which is trivial. Now suppose that (20) holds for some $g \leq \mathsf{lfp}(\psi)$. Then

$$\psi(g) \circ \nu_\emptyset(\sigma(t_1, \ldots, t_n)) = (F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle g, \mathsf{id}\rangle \vee \bigvee_{d' \in \Delta^E} g \circ \mu_\emptyset \circ d'_{T\emptyset})(\sigma(t_1, \ldots, t_n)) \,.$$

But since the equations are in the cfsc format, we have

$$F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle g, \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) = \bot\,. \tag{21}$$

Moreover, again by the cfsc format, $\sigma(t_1, \ldots, t_n)$ does not occur in any equation other than the defining one in E, and thus for all $d' \in \Delta^E$ with $d' \neq d$ we have

$$g \circ \mu_\emptyset \circ d'_{T\emptyset}(\sigma(t_1, \ldots, t_n)) = g \circ \nu_\emptyset(\sigma(t_1, \ldots, t_n))$$

which is below $\mathsf{lfp}\,(\psi) \circ \mu_\emptyset \circ d_{T\emptyset}(\sigma(t_1, \ldots, t_n))$ by the induction hypothesis (20). Together with the assumption that $g \leq \mathsf{lfp}\,(\psi)$ this implies

$$\bigvee_{d' \in \Delta^E} g \circ \mu_\emptyset \circ d'_{T\emptyset}(\sigma(t_1, \ldots, t_n)) \leq \mathsf{lfp}\,(\psi) \circ \mu_\emptyset \circ d_{T\emptyset}(\sigma(t_1, \ldots, t_n))\,.$$

By the above and (21), we may conclude

$$\psi(g) \circ \nu_\emptyset(\sigma(t_1, \ldots, t_n)) \leq \mathsf{lfp}\,(\psi) \circ \mu_\emptyset \circ d_{T\emptyset}(\sigma(t_1, \ldots, t_n))$$

as desired. This concludes the successor step; the limit step is again trivial (i.e., if we assume that (20) holds for a family of functions, then it also holds for the join of these functions). □

The following lemma is the main step for the correctness of the encoding of equations as assignment rules in Construction 5.8.

Lemma 5.11. *Let E and ψ be as above. If $t \equiv_E u$ then $Fq \circ (\mathsf{lfp}\,(\psi))(t) = Fq \circ (\mathsf{lfp}\,(\psi))(u)$, where q is the quotient map of $\equiv_E$.*

Proof. The proof is by induction on $\equiv_E$, that is, we show that the set of pairs $t \equiv_E u$ that satisfy $Fq \circ (\mathsf{lfp}\,(\psi))(t) = Fq \circ (\mathsf{lfp}\,(\psi))(u)$ is closed under each of the defining rules of $\equiv_E$. For reflexivity, transitivity and symmetry this is easy. The important cases are the two types of cfsc equations from E, and congruence.

For a σx-equation $\sigma_1(t_1, \ldots, t_n) \equiv_E \sigma_2(u_1, \ldots, u_n)$, by definition of Δ^E there is an assignment rule d such that $\mu_\emptyset \circ d_{T\emptyset}(\sigma_1(t_1, \ldots, t_n)) = \sigma_2(u_1, \ldots, u_n)$, and by definition of $\mathsf{lfp}\,(\psi)$ we have $\mathsf{lfp}\,(\psi) \circ \mu_\emptyset \circ d_{T\emptyset} \leq \mathsf{lfp}\,(\psi)$; so

$$(\mathsf{lfp}\,(\psi))(\sigma_2(u_1, \ldots, u_n)) \leq (\mathsf{lfp}\,(\psi))(\sigma_1(t_1, \ldots, t_n))\,.$$

For the converse inequality, there is another assignment rule d', and thus we also have $(\mathsf{lfp}\,(\psi))(\sigma_1(t_1, \ldots, t_n)) \leq (\mathsf{lfp}\,(\psi))(\sigma_2(u_1, \ldots, u_n))$.

For a defining equation $\sigma(t_1, \ldots, t_n) \equiv_E t$ we have a natural transformation in d such that $\mu_\emptyset \circ d_{T\emptyset}(\sigma(t_1, \ldots, t_n)) = t$. Thus $(\mathsf{lfp}\,(\psi))(t) = (\mathsf{lfp}\,(\psi)) \circ \mu_\emptyset \circ d_{T\emptyset}(\sigma(t_1, \ldots, t_n)) \leq (\mathsf{lfp}\,(\psi))(\sigma(t_1, \ldots, t_n))$. The converse inequality follows by Lemma 5.10. So $(\mathsf{lfp}\,(\psi))(t) = (\mathsf{lfp}\,(\psi))(\sigma(t_1, \ldots, t_n))$.

Finally, for the congruence rule, suppose there are terms $t_1, \ldots, t_n, u_1, \ldots, u_n$ such that $t_i \equiv_E u_i$ and $Fq \circ (\mathsf{lfp}\,(\psi))(t_i) = Fq \circ (\mathsf{lfp}\,(\psi))(u_i)$ for all $i \leq n$, and σ is an operator of arity n. This implies

$$\langle Fq \circ \mathsf{lfp}\,(\psi), q\rangle(t_i) = \langle Fq \circ \mathsf{lfp}\,(\psi), q\rangle(u_i) \qquad \text{for all } i \leq n \tag{22}$$

since $q(t_i) = q(u_i)$ for each i. Now

$$\begin{aligned}
&Fq \circ (\mathsf{lfp}\,(\psi))(\sigma(t_1, \ldots, t_n)) \\
&= Fq \circ F\mu_\emptyset \circ (\mathsf{lfp}\,(\varphi))_{T\emptyset} \circ \Sigma\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) \\
&= F\mu' \circ FTq \circ (\mathsf{lfp}\,(\varphi))_{T\emptyset} \circ \Sigma\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) \\
&= F\mu' \circ (\mathsf{lfp}\,(\varphi))_{(T\emptyset)/\equiv_E} \circ \Sigma(Fq \times q) \circ \Sigma\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) \\
&= F\mu' \circ (\mathsf{lfp}\,(\varphi))_{(T\emptyset)/\equiv_E} \circ \Sigma\langle Fq \circ \mathsf{lfp}\,(\psi), q\rangle(\sigma(t_1, \ldots, t_n)) \\
&= F\mu' \circ (\mathsf{lfp}\,(\varphi))_{(T\emptyset)/\equiv_E} \circ \Sigma\langle Fq \circ \mathsf{lfp}\,(\psi), q\rangle(\sigma(u_1, \ldots, u_n)) \\
&= Fq \circ F\mu_\emptyset \circ (\mathsf{lfp}\,(\varphi))_{T\emptyset} \circ \Sigma\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle(\sigma(u_1, \ldots, u_n)) \\
&= Fq \circ (\mathsf{lfp}\,(\psi))(\sigma(u_1, \ldots, u_n))
\end{aligned}$$

where the first equality holds by Theorem 4.14, the second since q is an algebra morphism, the third by naturality of $(\mathsf{lfp}\,(\varphi))$, the fourth by functoriality, the fifth by the induction hypothesis, and the last two equalities are as before.

Notice that we used the fact that the quotient map q is an algebra morphism to some T-algebra μ'. It is worthwhile to note that we need to reason up to $\equiv_E$ to get (22). Indeed, $\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle(t_i) = \langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle(u_i)$ does not hold in general, since t_i is only congruent to u_i, not necessarily equal. □

This allows to prove that $\mathsf{lfp}\,(\psi)$ and $\mathsf{lfp}\,(\theta)$ coincide "up to $\equiv_E$".

Lemma 5.12. *Let ψ and q be as above. Then $Fq \circ (\mathsf{lfp}\,(\theta)) = Fq \circ (\mathsf{lfp}\,(\psi))$.*

Proof. We first prove that $\psi(\mathsf{lfp}\,(\theta)) \leq \mathsf{lfp}\,(\theta)$. By definition of θ we have $F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle \mathsf{lfp}\,(\theta), \mathsf{id}\rangle \leq \mathsf{lfp}\,(\theta) \circ \nu_\emptyset$. So the interesting part is to show that $\mathsf{lfp}\,(\theta) \circ \mu_\emptyset \circ d_{T\emptyset} \leq \mathsf{lfp}\,(\theta) \circ \nu_\emptyset$ for any $d \in \Delta^E$, given that $\bigvee_{r\in R} \mathsf{lfp}\,(\theta) \circ r \circ q \circ \nu_\emptyset \leq \mathsf{lfp}\,(\theta) \circ \nu_\emptyset$ (which holds since $\mathsf{lfp}\,(\theta)$ is a fixed point of θ). But this is simple, given that each d acts on an argument either as the identity or by an equation in E. Thus $\psi(\mathsf{lfp}\,(\theta)) \leq \mathsf{lfp}\,(\theta)$; by (fixed point) induction we then have $\mathsf{lfp}\,(\psi) \leq \mathsf{lfp}\,(\theta)$, and thus $Fq \circ \mathsf{lfp}\,(\psi) \leq Fq \circ \mathsf{lfp}\,(\theta)$.

We proceed to show $Fq \circ \mathsf{lfp}\,(\theta) \leq Fq \circ \mathsf{lfp}\,(\psi)$ by transfinite induction; the main step is to prove that $Fq \circ h \leq Fq \circ \mathsf{lfp}\,(\psi)$ implies $Fq \circ \theta(h) \leq Fq \circ \mathsf{lfp}\,(\psi)$. So suppose $Fq \circ h \leq Fq \circ \mathsf{lfp}\,(\psi)$. Then

$$\begin{aligned}
Fq \circ \theta(h) \circ \nu_\emptyset &= Fq \circ (F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle h, \mathsf{id}\rangle \vee \bigvee_{r\in R} h \circ r \circ q \circ \nu_\emptyset) \\
&= Fq \circ F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle h, \mathsf{id}\rangle \vee \bigvee_{r\in R} Fq \circ h \circ r \circ q \circ \nu_\emptyset
\end{aligned}$$

Now

$$\begin{aligned}
Fq \circ F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle h, \mathsf{id}\rangle &= F\mu' \circ FTq \circ \rho_{T\emptyset} \circ \Sigma\langle h, \mathsf{id}\rangle \\
&= F\mu' \circ \rho_{(T\emptyset)/\equiv_E} \circ \Sigma(Fq \times q) \circ \Sigma\langle h, \mathsf{id}\rangle \\
&\leq F\mu' \circ \rho_{(T\emptyset)/\equiv_E} \circ \Sigma(Fq \times q) \circ \Sigma\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle \\
&= Fq \circ F\mu_\emptyset \circ \rho_{T\emptyset} \circ \Sigma\langle \mathsf{lfp}\,(\psi), \mathsf{id}\rangle \\
&\leq Fq \circ \mathsf{lfp}\,(\psi) \circ \nu_\emptyset
\end{aligned}$$

where μ' is the algebra structure induced by q. The first inequality holds by assumption ($Fq \circ h \leq Fq \circ \mathsf{lfp}\,(\psi)$) and the second one by the fact that $\mathsf{lfp}\,(\psi)$ is a fixed point of ψ and by monotonicity of Fq. Moreover

$$\bigvee_{r \in R} Fq \circ h \circ r \circ q \circ \nu_\emptyset \leq \bigvee_{r \in R} Fq \circ \mathsf{lfp}\,(\psi) \circ r \circ q \circ \nu_\emptyset = Fq \circ \mathsf{lfp}\,(\psi) \circ \nu_\emptyset$$

where the inequality holds by assumption. For the equality, recall that R is the set of right inverses of q, so that for any $t \in T\emptyset$ and any $r \in R$, we have $r \circ q(t) \equiv_E t$. By Lemma 5.11 we then obtain $Fq \circ (\mathsf{lfp}\,(\psi)) \circ r \circ q(t) = Fq \circ (\mathsf{lfp}\,(\psi))(t)$.

We conclude that $Fq \circ \theta(h) \leq Fq \circ \mathsf{lfp}\,(\psi)$ as desired. □

This implies that $\mathsf{lfp}\,(\theta)$ and $\mathsf{lfp}\,(\psi)$ are *behaviourally equivalent* up to $\equiv_E$. It is well-known that behavioural equivalence coincides with bisimilarity whenever the functor F preserves weak pullbacks [37], a mild condition satisfied by most functors used in practice, including, e.g., transition systems and stream systems. Under this assumption one can prove that $\mathsf{lfp}\,(\theta)$ is equal to $\mathsf{lfp}\,(\psi)$ up to bisimilarity, and by Theorem 4.14 we then obtain our main result of this section.

Theorem 5.13. *Suppose E is a set of equations which is in the* cfsc *format w.r.t. ρ, and suppose the behaviour functor F preserves weak pullbacks. Then the interpretation of ρ and E equals the operational model of some abstract GSOS specification, up to bisimilarity. Bisimilarity is a congruence, and $bis \circ ctx \circ bis$ is $b_{\langle \mathsf{lfp}\,(\theta), \mathsf{id} \rangle}$-compatible.*

Proof. Consider the following diagram.

$$\begin{array}{ccccc} \equiv_E & \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} & T\emptyset & \xrightarrow{\;q\;} & (T\emptyset)/{\equiv_E} \\ & & \Big\downarrow{\scriptstyle \mathsf{lfp}\,(\psi)} & & \Big\downarrow \\ & & FT\emptyset & \xrightarrow[Fq]{} & F(T\emptyset)/{\equiv_E} \end{array}$$

Lemma 5.11 shows that $Fq \circ (\mathsf{lfp}\,(\psi))$ is well-defined on equivalence classes in $(T\emptyset)/{\equiv_E}$. Hence there is a (unique) dashed arrow as in the diagram, making the square commute. This turns $(T\emptyset)/{\equiv_E}$ into a coalgebra, and q into a coalgebra homomorphism.

Further, by Lemma 5.12, q is also a homomorphism from $\mathsf{lfp}\,(\theta)$ to the same coalgebra. Now the pullback of q along itself is simply $\equiv_E$, and since F preserves weak pullbacks, $\equiv_E$ is a bisimulation between $\mathsf{lfp}\,(\psi)$ and $\mathsf{lfp}\,(\theta)$ [37]. Thus, in particular, $\mathsf{lfp}\,(\psi)$ and $\mathsf{lfp}\,(\theta)$ are equal up to bisimilarity, since $\equiv_E$ is reflexive.

By Theorem 4.14, $\mathsf{lfp}\,(\psi)$ is the model of a certain abstract GSOS specification. Hence bisimilarity is a congruence on $\mathsf{lfp}\,(\psi)$, and since $\mathsf{lfp}\,(\psi)$ and $\mathsf{lfp}\,(\theta)$ are equal up to bisimilarity, it follows from Lemma 2.3 that bisimilarity is a congruence on $\mathsf{lfp}\,(\theta)$. Finally, again since $\mathsf{lfp}\,(\psi)$ is the model of an abstract GSOS specification, ctx is $b_{\langle \mathsf{lfp}\,(\psi), \mathsf{id} \rangle}$-compatible. Thus, by Lemma 2.4, $bis \circ ctx \circ bis$ is $b_{\langle \mathsf{lfp}\,(\theta), \mathsf{id} \rangle}$-compatible. □

6. Related Work

The main work in the literature that treats the meta-theory of rule formats with structural congruences [30] focuses on labelled transition systems, whereas our results apply to coalgebras in general (for behaviour functors with a complete lattice structure). Concerning transition systems, the basic rule format in [30] is tyft/tyxt[2], which is more expressive than positive GSOS since it allows lookahead in the premises. However, while [30] proves congruence of bisimilarity this does not imply the compatibility (or even soundness) of bisimulation up to context [32], which we obtain in the present work (and which is, in fact, problematic in the presence of lookahead).

Plotkin proposed to model recursion by interpreting abstract GSOS in the category of complete partial orders [31]. Klin [20] showed that by moving to categories enriched in complete partial orders, one can interpret recursive constructs which have a similar form as our assignment rules. Technically our approach is different as it is based on an order on the behaviour functor, rather than interpreting everything in an ordered setting and using an infinite unfolding of terms, as is done in [20]. Further, in [20] each operator is either specified by an equation or by operational rules, disallowing a specification such as that of the parallel composition in Equation (2).

In [27], various constructions on distributive laws are presented. Example 32 of that paper discusses the definition of the parallel composition as in (2) above, but a general theory for structural congruence is missing. Distributive laws are applied in [17] to find solutions of guarded recursive equations. Further, in [28], recursive equations are interpreted in the context of iterative algebras, where operations of interest are given by an abstract GSOS specification. That work seems to focus mainly on solutions to guarded equations, but the precise connection to the present work remains to be understood.

In [6], it is shown how to obtain a distributive law for a monad that is the quotient of another one by imposing extra equations, under the condition that the distributive law respects the equations. However, this condition requires that the equations already hold semantically, which is fundamentally different from the present paper where we define behaviour by imposing equations on an operational specification. Similarly, in [8, 9], it is shown how to lift calculi with structural axioms to coalgebraic models, but under the assumption, again, that the equations already hold.

7. Conclusions

We extended Turi and Plotkin's bialgebraic approach to operational semantics with non-structural assignment rules and structural congruence, providing a general coalgebraic framework for monotone abstract GSOS with equations.

[2] In [30], it is sketched how to extend the results to the ntyft/ntyxt, which involves however a complicated integration of the cfsc format with the notion of stable model.

Technically, our results are based on the combination of bialgebraic semantics with order. Our main result is that the interpretation of a specification involving assignment rules is well-behaved, in the sense that bisimilarity is a congruence and bisimulation-up-to techniques are sound. This result carries over to specifications with structural congruence in the cfsc format proposed in [30].

There are several directions for future work. First, our techniques can possibly be extended to allow lookahead in premises by using cofree comonads (see, e.g., [23]). While in general the combined use of cofree comonads and free monads in specifications is known to be problematic [24], we expect that part of these problems may be addressed by considering only positive (monotone) specifications. In fact, this could form the basis for a bialgebraic account of the tyft format. Second, in the current work we only consider free monads. One may incorporate equations which already hold, for instance by using the theory of [6].

At a more fundamental level, we believe that the combination of bialgebraic semantics with ordered structures is an interesting direction of research which is yet to be explored further (cf. [11, 5]). In the current paper, we used this theory only in a relatively concrete manner, by focusing on Set functors and specifications where the syntax is given by a signature. A more abstract categorical perspective, for instance in terms of order enriched categories, could potentially generalize some of this technical development. Such a generalization could be of interest, for instance, to study structural congruences for calculi with names.

Acknowledgments

We are grateful to Daniel Gebler, Bartek Klin, Mohammad Reza Mousavi and Jan Rutten for helpful discussions, and to the anonymous referees for numerous valuable comments and suggestions. Our research has been funded by the Netherlands Organisation for Scientific Research (NWO), CoRE project, dossier number: 612.063.920. Part of the research of the first author was carried out during his stay at Leiden University.

References

[1] L. Aceto, W. Fokkink, and C. Verhoef. Structural operational semantics. In *Handbook of Process Algebra*, pages 197–292. Elsevier, 2001.

[2] M. Barr. Coequalizers and free triples. *Math. Z.*, 116(4):307–322, 1970.

[3] F. Bartels. *On generalised coinduction and probabilistic specification formats*. PhD thesis, CWI, Amsterdam, 2004.

[4] B. Bloom, S. Istrail, and A. Meyer. Bisimulation can't be traced. *J. of ACM*, 42(1):232–268, 1995.

[5] F. Bonchi, D. Petrisan, D. Pous, and J. Rot. Lax bialgebras and up-to techniques for weak bisimulations. In L. Aceto and D. de Frutos-Escrig, editors, *Proc. of 26th International Conference on Concurrency Theory,*

CONCUR 2015, volume 42 of *Leibniz Int. Proc. in Inf.*, pages 240–253. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[6] M. Bonsangue, H. Hansen, A. Kurz, and J. Rot. Presenting distributive laws. *Log. Meth. in Comput. Sci.*, 11(3:2), 2015.

[7] P. Buchholz and P. Kemper. Quantifying the dynamic behavior of process algebras. In L. de Alfaro and S. Gilmore, editors, *Proc. of Process Algebra and Probabilistic Methods, Performance Modeling and Verification: Joint International Workshop, PAPM-PROBMIV 2001*, volume 2165 of *Lect. Notes in Comput. Sci.*, pages 184–199. Springer, 2001.

[8] M. G. Buscemi and U. Montanari. A first order coalgebraic model of pi-calculus early observational equivalence. In L. Brim, P. Jancar, M. Kretínský, and A. Kucera, editors, *Proc. of Concurrency Theory, 13th International Conference, CONCUR 2002*, volume 2421 of *Lecture Notes in Computer Science*, pages 449–465. Springer, 2002.

[9] A. Corradini, R. Heckel, and U. Montanari. Compositional SOS and beyond: a coalgebraic view of open systems. *Theor. Comput. Sci.*, 280(1-2):163–192, 2002.

[10] M. Droste and W. Kuich. Semirings and formal power series. In Manfred Droste, Werner Kuich, and Heiko Vogler, editors, *Handbook of Weighted Automata*, pages 3–28. Springer, 2009.

[11] M. Fiore and S. Staton. Positive structural operational semantics and monotone distributive laws. In *CMCS Short Contributions*, pages 8–9, 2010. Tech. Report No. SEN-1004, CWI.

[12] G. Gierz, K.H. Hofmann, K. Keimel, J.D. Lawson, M. Mislove, and D.S. Scott. *Continuous Lattices and Domains*, volume 93 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, 2003.

[13] R. van Glabbeek. The meaning of negative premises in transition system specifications II. *J. of Log. and Algebr. Program.*, 60-61:229–258, 2004.

[14] C. Hermida and B. Jacobs. Structural induction and coinduction in a fibrational setting. *Inf. and Comput.*, 145(2):107–152, 1998.

[15] P. Hitchcock and D. Park. Induction rules and termination proofs. In M. Nivat, editor, *Proc. of 1st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 225–251. North-Holland, 1972.

[16] J. Hughes and B. Jacobs. Simulations in coalgebra. *Theor. Comput. Sci.*, 327(1-2):71–108, 2004.

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

[18] B. Jacobs. Introduction to coalgebra. Towards mathematics of states and observations. Version 2.0. `http://www.cs.ru.nl/B.Jacobs/CLG/JacobsCoalgebraIntro.pdf`, 2012.

[19] B. Jacobs and J. Rutten. An introduction to (co)algebras and (co)induction. In D. Sangiorgi and J. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, volume 52 of *Cambridge Tracts in Theoretical Computer Science*, pages 38–99. Cambridge University Press, 2012.

[20] B. Klin. Adding recursive constructs to bialgebraic semantics. *J. of Log. and Algebr. Program.*, 60-61:259–286, 2004.

[21] B. Klin. Bialgebraic methods in structural operational semantics: Invited talk. *Electron. Notes in Theor. Comput. Sci.*, 175(1):33–43, 2007.

[22] B. Klin. Structural operational semantics for weighted transition systems. In J. Palsberg, editor, *Semantics and Algebraic Specification*, volume 5700 of *Lect. Notes in Comput. Sci.*, pages 121–139. Springer, 2009.

[23] B. Klin. Bialgebras for structural operational semantics: An introduction. *Theor. Comput. Sci.*, 412(38):5043–5069, 2011.

[24] B. Klin and B. Nachyla. Distributive laws and decidable properties of SOS specifications. In J. Borgström and S. Crafa, editors, *Proc. of Combined 21st International Workshop on Expressiveness in Concurrency and 11th Workshop on Structural Operational Semantics, EXPRESS/SOS 2014*, volume 160 of *Electron. Proc. in Theor. Comput. Sci.*, pages 79–93. Open Publishing Association, 2014.

[25] C. Kupke, M. Niqui, and J. Rutten. Stream differential equations: concrete formats for coinductive definitions. Tech. Report No. RR-11-10, Oxford University, 2011.

[26] J. Lambek. A fixpoint theorem for complete categories. *Math. Z.*, 103(2):151–161, 1968.

[27] M. Lenisa, J. Power, and H. Watanabe. Category theory for operational semantics. *Theor. Comput. Sci.*, 327(1-2):135–154, 2004.

[28] S. Milius, L. Moss, and D. Schwencke. Abstract GSOS rules and a modular treatment of recursive definitions. *Log. Meth. in Comput. Sci.*, 9(3:28):52 pp., 2013.

[29] R. Milner. Functions as processes. *Math. Struct. in Comput. Sci.*, 2(2):119–141, 1992.

[30] M. R. Mousavi and M. A. Reniers. Congruence for structural congruences. In V. Sassone, editor, *Proc. of Foundations of Software Science and Computational Structures, 8th International Conference, FoSSaCS 2005*, volume 3441 of *Lect. Notes in Comput. Sci.*, pages 47–62. Springer, 2005.

[31] G. D. Plotkin. Bialgebraic semantics and recursion (extended abstract). *Electron. Notes in Theor. Comput. Sci.*, 44(1):285–288, 2001.

[32] D. Pous and D. Sangiorgi. Enhancements of the bisimulation proof method. In D. Sangiorgi and J. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, volume 52 of *Cambridge Tracts in Theoretical Computer Science*, pages 233–289. Cambridge University Press, 2012.

[33] J. Rot, F. Bonchi, M. Bonsangue, D. Pous, J. Rutten, and A. Silva. Enhanced coalgebraic bisimulation. *Math. Struct. in Comput. Science*, 2016. To appear, doi:10.1017/s0960129515000523.

[34] J. Rot, M. Bonsangue, and J. Rutten. Coalgebraic bisimulation-up-to. In P. van Emde Boas et al., editor, *Proc. of 39th International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2013*, volume 7741 of *Lect. Notes in Comput. Sci.*, pages 369–381. Springer, 2013.

[35] J. Rot and M. M. Bonsangue. Combining bialgebraic semantics and equations. In A. Muscholl, editor, *Proc. of Foundations of Software Science and Computation Structures - 17th International Conference, FoSSaCS 2014*, volume 8412 of *Lect. Notes in Comput. Sci.*, pages 381–395. Springer, 2014.

[36] J. Rutten. Relators and metric bisimulations. *Electron. Notes in Theor. Comput. Sci.*, 11:252–258, 1998.

[37] J. Rutten. Universal coalgebra: a theory of systems. *Theor. Comput. Sci.*, 249(1):3–80, 2000.

[38] J. Rutten. Behavioural differential equations: a coinductive calculus of streams, automata, and power series. *Theor. Comput. Sci.*, 308(1-3):1–53, 2003.

[39] D. Sangiorgi. On the bisimulation proof method. *Math. Struct. in Comput. Sci.*, 8(5):447–479, 1998.

[40] D. Sangiorgi. *An Introduction to Bisimulation and Coinduction*. Cambridge University Press, 2012.

[41] D. Sangiorgi and D. Walker. *The Pi-Calculus - a Theory of Mobile Processes*. Cambridge University Press, 2001.

[42] D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *Proc. of 12th Annual IEEE Symposium on Logic in Computer Science, LICS 1997*, pages 280–291. IEEE Computer Society, 1997.