Enhanced Coinduction

Rot, J.C.

Chapter 6

Bialgebraic semantics with equations

In this chapter, we focus on structural operational semantics, in the setting of abstract GSOS specifications as introduced by Turi and Plotkin. As explained in Section 3.5 and the introduction, their approach provides a general perspective on well-behaved, compositional calculi and languages, parametric in the type of behaviour and the type of syntax. Moreover, in the previous chapters we have seen that bisimulation up to context is a sound (even compatible) proof technique on models of abstract GSOS specifications.

Given a GSOS specification, the behaviour of terms 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:

$$\frac{!x|x \xrightarrow{a} t}{!x \xrightarrow{a} t} \tag{6.1}$$

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 operational model on closed terms is then defined as the *least* model such that every transition can either can be derived from a rule in the specification or from

[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 $!(a.P + \overline{a}.Q)$ [PS12, SW01].

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 (see Section 5.4.1).

The main result of this chapter is that the interpretation of a monotone abstract GSOS specification together with a set of assignment rules is itself the operational 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 [PS12]. 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., [PS12]).

In the second part of this chapter, 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 [Mil92]. 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:

$$\frac{x \xrightarrow{a} x'}{x|y \xrightarrow{a} x'|y} \qquad x|y = y|x \qquad\qquad !x = !x|x \tag{6.2}$$

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 [MR05]. 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, for which bisimilarity is a congruence when combined with tyft specifications.

In the current chapter, 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 when 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 [Bar04, Kli07]. 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 In Section 6.1, we introduce assignment rules and their interpretation. In Section 6.2, we show that this interpretation can be obtained as the operational model of another abstract GSOS specification. Section 6.3 contains the integration of structural congruence with the bialgebraic framework. In Section 6.4, we conclude and discuss related work.

6.1 Assignment rules

We consider the interpretation of abstract GSOS specifications (without negative premises) together with *assignment rules* of the form

$$\sigma(x_1, \ldots, x_n) := t \tag{6.3}$$

where t is a term over the variables $x_1, \ldots, x_n$. Assignment rules will be interpreted as a kind of rewriting rules: the behaviour of t induces behaviour of $\sigma(x_1, \ldots, x_n)$. An example is the replication operator given in equation (6.1) 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 infinite inferences, one is interested in the *least* transition system on closed terms which is a model in the above sense. Such a least model does not necessarily exist in general because of negative premises. Therefore, we will restrict to GSOS specifications without negative premises.

To interpret specifications which involve assignment rules at the general level of a functor $B\colon \mathsf{Set} \to \mathsf{Set}$ one needs a notion of order on B. In the case of labelled transition systems, this order is clear and often left implicit: in that case $BX = (\mathcal{P}X)^A$, and the order is simply the (pointwise) subset order. To allow the desired generalization, we assume that our behaviour functor B is ordered (cf. Section 5.4.1). We will need the existence of fixed points of monotone functions.

To this end, let CJSL be the category of complete (join semi-)lattices and join-preserving functions. We define a CJSL-*ordered functor* to be a functor $B\colon \mathsf{Set} \to \mathsf{Set}$ with a factorization $\sqsubseteq$ through CJSL:

$$\begin{tikzcd} & \mathsf{CJSL} \arrow[d, "U"] \\ \mathsf{Set} \arrow[ur, "\sqsubseteq"] \arrow[r, "B"'] & \mathsf{Set} \end{tikzcd}$$

where U is the forgetful functor. If B is a CJSL-ordered functor, then for any set X, BX is a complete lattice. We denote the join of a set $S \subseteq BX$ in this lattice by $\bigvee S$, and we write $\perp$ for $\bigvee \emptyset$ and $x \leq y$ if $x \vee y = y$, for $x, y \in BX$. Moreover, for any function $f\colon X \to Y$, Bf is join-preserving. Consequently, Bf is also monotone, i.e., for any $x, y \in BX$: $x \leq y$ implies $(Bf)(x) \leq (Bf)(y)$.

Example 6.1.1. The functor $(\mathcal{P}-)^A$ of labelled transition systems is CJSL-ordered, with the order on $(\mathcal{P}X)^A$ given by pointwise subset inclusion.

Example 6.1.2. In Chapter 3 we defined weighted transition systems for a semiring as coalgebras for the functor $(\mathcal{M}-)^A$, where $\mathcal{M}X$ consists of (finite) linear combinations with coefficients in the semiring. Here, we consider weighted transition systems for a *complete monoid* M, i.e., a monoid with an infinitary sum operation consistent with the finite sum [DK09]. These 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 $\varphi \in M^X$ into $\varphi^h \in M^Y$ defined, for all $y \in Y$, by $\varphi^h(y) = \sum_{x' \in h^{-1}(y)} \varphi(x')$.

By taking the Boolean monoid, we retrieve infinitely branching labelled transition systems. As another example, consider the set $\mathbb{R}^+ \cup \{\infty\}$ of positive reals, ordered as usual and extended with a top element ∞. Together with the supremum operation, $\mathbb{R}^+ \cup \{\infty\}$ forms a complete ordered monoid, with 0 as unit. The order on $\mathbb{R}^+ \cup \{\infty\}$ extends to an order on the functor for weighted transition systems over this monoid, where joins are calculated pointwise.

Example 6.1.3. For a non-example: we can try to extend a functor $B\colon \mathsf{Set} \to \mathsf{Set}$ to a CJSL-ordered functor B' by defining $B'X = BX + 2$, putting the discrete order on BX and taking the elements of $2 = \{\top, \perp\}$ to be the top and the bottom element respectively. Contrary to what is stated in [RB14, Example 2], such a functor B' is not CJSL-ordered, in general. Indeed $B'X$ is a complete lattice, but the functor B' is not well-defined on morphisms: given a function f, $B'f$ need not be join-preserving. For instance, if we take $B = \mathsf{Id}$, 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 $(B'f)(x) \vee (B'f)(y) = (B'f)(x) = f(x) \neq \top$ whereas $(B'f)(x \vee y) = (B'f)(\top) = \top$.

Given arbitrary sets X and Y, the complete lattice on BY lifts pointwise to a complete lattice on functions of type $X \to BY$, i.e., for a collection $\{f_i\}_{i \in I}$ of functions of the form $f_i \colon X \to BY$ 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 over a signature. Given a polynomial functor $\Sigma \colon \mathsf{Set} \to \mathsf{Set}$ corresponding to a signature (Section 3.4), we denote this set by

$$\mathbb{M} = \{f \mid f \colon \Sigma^*\emptyset \to B\Sigma^*\emptyset\}\,. \tag{6.4}$$

The order on B lifts to an order on $B \times \mathsf{Id}$ by defining $(b_1, x_1) \leq (b_2, x_2)$ iff $b_1 \leq b_2$ and $x_1 = x_2$ for $(b_1, x_1), (b_2, x_2) \in BX \times X$. Moreover, given Σ as above, the order lifts componentwise to ΣBX (and also to $\Sigma(BX \times X)$) for any set X, by defining, for any operators σ, τ 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 6.1.4. Using the above lifting of the order on B to $\Sigma(B \times \mathsf{Id})$, a specification $\rho \colon \Sigma(B \times \mathsf{Id}) \Rightarrow B\Sigma^*$ is said to be *monotone* if all its components are monotone.

Definition 6.1.4 is a special case of monotone abstract GSOS specifications defined in terms of relation lifting, as introduced in Section 5.4.1. For the functor $BX = (\mathcal{P}X)^A$ of labelled transition systems, monotone specifications correspond to specifications in (an infinitary version of) the *positive GSOS* format [FS10].

Assignment rules (6.3) can be formalized in terms of natural transformations, which are independent of the behaviour functor B.

Definition 6.1.5. An *assignment rule* is a natural transformation $d \colon \Sigma \Rightarrow \Sigma^*$.

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. For example, the assignment rule for the replication operator is the natural transformation that sends $!x$ to $!x|x$ for any x, and is the identity on all other operators in Σ.

Assumption 6.1.6. In the remainder of this chapter, we assume:

1. A CJSL-ordered functor B.
2. A functor Σ defined from a signature (see Section 3.4), with free monad (Σ^*, η, μ).
3. A monotone GSOS specification $\rho \colon \Sigma(B \times \mathsf{Id}) \Rightarrow B\Sigma^*$.
4. A set Δ of assignment rules, ranged over by $d \colon \Sigma \Rightarrow \Sigma^*$.

Throughout this chapter we denote by $M(\rho)$ the *operational model* of ρ. As explained in Section 3.5.2, the operational model $M(\rho) \colon \Sigma^*\emptyset \to B\Sigma^*\emptyset$ is the unique coalgebra that makes the following diagram commute:

$$
\begin{tikzcd}
\Sigma\Sigma^*\emptyset \arrow[r, "\Sigma\langle M(\rho),\mathsf{id}\rangle"] \arrow[dd, "\kappa_\emptyset"'] & \Sigma(B\Sigma^*\emptyset \times \Sigma^*\emptyset) \arrow[d, "\rho_{\Sigma^*\emptyset}"] \\
 & B\Sigma^*\Sigma^*\emptyset \arrow[d, "B\mu_\emptyset"] \\
\Sigma^*\emptyset \arrow[r, "M(\rho)"'] & B\Sigma^*\emptyset
\end{tikzcd}
\tag{6.5}
$$

where $\kappa\colon \Sigma\Sigma^* \Rightarrow \Sigma^*$ is the natural transformation such that, for a component X, the copairing $[\kappa_X, \eta_X]$ is the initial $\Sigma + X$-algebra (Equation (3.11) in Section 3.4). Observe that the operational model is the unique $f \in \mathbb{M}$ (see Equation 6.4) satisfying the equation

$$f \circ \kappa_\emptyset = B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \,.$$

The definition below extends this equation to incorporate assignment rules.

Definition 6.1.7. Let $\psi\colon \mathbb{M} \to \mathbb{M}$ be the (unique) function such that

$$\psi(f) \circ \kappa_\emptyset = B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{d\in\Delta} f \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset} \,.$$

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

The function ψ is indeed uniquely defined, since $\kappa_\emptyset\colon \Sigma\Sigma^*\emptyset \to \Sigma^*\emptyset$ is an initial algebra and therefore an isomorphism. 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 intended interpretation. In order to show that a least model exists, we need the following.

Lemma 6.1.8. *The function* $\psi\colon \mathbb{M} \to \mathbb{M}$ *is monotone.*

Proof. Let $f, g \in \mathbb{M}$ with $f \leq g$. By monotonicity of ρ, we have $\rho_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq \rho_{\Sigma^*\emptyset} \circ \Sigma\langle g, \mathsf{id}\rangle$, and since $B\mu_\emptyset$ is monotone then $B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle g, \mathsf{id}\rangle$. It follows that $\psi(f) \circ \kappa_\emptyset \leq \psi(g) \circ \kappa_\emptyset$ and thus also $\psi(f) \leq \psi(g)$ because $\kappa_\emptyset$ 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 6.1.9. The *interpretation* of ρ and Δ is the least (ρ, Δ)-model, i.e., $\mathsf{lfp}(\psi)$.

Example 6.1.10. 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 $t \xrightarrow{a} t'$.

This is a recursive definition; being the least such transition system has the desired consequence that every derivation of a transition $t \xrightarrow{a} t'$ is finite.

6.2 Integrating assignment rules in abstract GSOS

In the previous section, we have seen how to interpret a monotone 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 the 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 $\mathbb{G}$ be the set of all monotone abstract GSOS specifications of Σ over B (Definition 6.1.4). We turn $\mathbb{G}$ into a complete lattice by defining the order componentwise, i.e., for any $L \subseteq \mathbb{G}$ and any set X: $(\bigvee L)_X = \bigvee_{\rho \in L} \rho_X$. The join is well-defined:

Lemma 6.2.1. *For any $L \subseteq \mathbb{G}$: 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 $k \in \Sigma(BX \times X)$:

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

which proves naturality. Monotonicity is straightforward as well. $\square$

The lattice structure of $\mathbb{G}$ provides a way of combining specifications. Consider, for an assignment rule $d \in \Delta$ and specification τ, the following natural transformation:

$$\Sigma(B \times \mathsf{Id}) \xrightarrow{d_{B \times \mathsf{Id}}} \Sigma^*(B \times \mathsf{Id}) \xrightarrow{\tau^\dagger} B\Sigma^* \times \Sigma^* \xrightarrow{\pi_1} B\Sigma^* \tag{6.6}$$

Recall from Section 3.5.2 that $\tau^\dagger$ 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 $k_1, \ldots, k_n \in BX \times X$ of its arguments, it first applies the assignment rule d to obtain a term $t(k_1, \ldots, k_n)$. Subsequently $\tau^\dagger$ is used to compute the behaviour of t given the behaviour $k_1, \ldots, k_n$. In short, the above transformation computes the behaviour of an operator by using rules from τ and a single application of the assignment rule d.

Example 6.2.2. Suppose the signature Σ contains a binary operator and a unary operator (to be interpreted as parallel composition | and replication ! respectively). Further, let ρ be a GSOS specification defined as usual for | (Example 3.5.4), and without any rules for the replication operator $!x$. Let d be the assignment rule associated to the replication, i.e., the identity on all operators except $!x$, which is mapped to $!x|x$.

Then the natural transformation in (6.6) corresponds to a specification in which there is a rule that concludes with $!x \to t$ for some t if and only if there is a derivation of $!x|x \to t$ in the GSOS specification ρ, from the same premises. Since there are no rules for $!x$ in ρ, the only possible derivation is

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

and therefore, the only rule for $!x$ is

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

The natural transformation in (6.6) is unchanged on all other operators.

As explained in the introduction of this chapter, this is not quite the correct specification of replication yet, but it is a first step. To obtain the correct specification, we need to apply such a construction recursively, which we will do below. First we define a function φ on $\mathbb{G}$ which uses the above construction to build, from an argument specification τ (of Σ over B), the specification containing all rules from the fixed specification ρ and all rules which can be formed as in (6.6).

Definition 6.2.3. Given our fixed ρ and Δ (Assumption 6.1.6), the map $\varphi\colon \mathbb{G} \to \mathbb{G}$ is defined as

$$\varphi(\tau) = \rho \vee \bigvee_{d \in \Delta} (\pi_1 \circ \tau^\dagger \circ d_{B \times \mathsf{Id}})\,.$$

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 $\tau^\dagger$, which requires an order on Σ^*. Any partial order $(X, \leq)$ inductively extends to an order on $\Sigma^* X$ 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 Section 5.4.1; here, we provide a concrete, self-contained exposition).

Lemma 6.2.4. *If τ is a monotone specification, then $\varphi(\tau)$ is monotone as well.*

Proof. We prove that if τ is monotone then the induced distributive law $\tau^\dagger\colon \Sigma^*(B \times \mathsf{Id}) \Rightarrow B\Sigma^* \times \Sigma^*$ is also monotone, by induction on pairs of terms $t, u \in \Sigma^*(BX \times X)$ with $t \le u$ (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 $(b,x),(c,y) \in BX \times X$ with $(b,x) \le (c,y)$ (so $b \le c$ and $x = y$) then

$$\tau_X^\dagger \circ \eta_{BX\times X}(b,x) = (B\eta_X \times \eta_X)(b,x) \le (B\eta_X \times \eta_X)(c,y) = \tau_X^\dagger \circ \eta_{BX\times X}(c,y)$$

where the inequality holds by monotonicity of $B\eta_X$ and since $x = y$, and the equalities by definition of $\tau^\dagger$ (Equation (3.15) in Section 3.5.2).

Suppose σ is an operator of arity n, and $t_1, \dots, t_n, u_1, \dots, u_n \in \Sigma^*(BX \times X)$ with $\tau_X^\dagger(t_i) \le \tau_X^\dagger(u_i)$ for all i. Then

$$\begin{array}{ll}
\tau_X^\dagger \circ \kappa_{BX\times X}(\sigma(t_1,\dots,t_n)) & \\
= (B\mu_X \times \kappa_X) \circ \langle \tau_{\Sigma^* X}, \Sigma\pi_2\rangle \circ \Sigma\tau_X^\dagger(\sigma(t_1,\dots,t_n)) & \text{definition } \tau^\dagger \\
= (B\mu_X \times \kappa_X) \circ \langle \tau_{\Sigma^* X}, \Sigma\pi_2\rangle(\sigma(\tau_X^\dagger(t_1),\dots,\tau_X^\dagger(t_n))) & \text{definition } \Sigma \\
\le (B\mu_X \times \kappa_X) \circ \langle \tau_{\Sigma^* X}, \Sigma\pi_2\rangle(\sigma(\tau_X^\dagger(u_1),\dots,\tau_X^\dagger(u_n))) & \text{see below} \\
= \tau_X^\dagger \circ \kappa_{BX\times X}(\sigma(u_1,\dots,u_n)) &
\end{array}$$

The inequality holds by monotonicity of $B\mu_X$ and τ, and the induction hypothesis; note that the induction hypothesis implies $\pi_2 \circ \tau_X^\dagger(t_i) = \pi_2 \circ \tau_X^\dagger(u_i)$ for all i. $\square$

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

Lemma 6.2.5. *The function $\varphi\colon \mathbb{G} \to \mathbb{G}$ is monotone.*

The main step in the proof of Lemma 6.2.5 is to show that the extension $(-)^\dagger$ of abstract GSOS specifications to distributive laws is monotone.

Lemma 6.2.6. *Let τ_1, τ_2 be specifications. If $\tau_1 \le \tau_2$ then $\pi_1 \circ (\tau_1^\dagger) \le \pi_1 \circ (\tau_2^\dagger)$.*

Proof. We have

$$(\tau_1^\dagger)_X \circ \eta_{BX\times X} = B\eta_X \times \eta_X = (\tau_2^\dagger)_X \circ \eta_{BX\times X}$$

by definition of $(-)^\dagger$ (Equation (3.15) in Section 3.5.2). Moreover

$$(B\mu_X \times \kappa_X) \circ \langle (\tau_1)_{\Sigma^* X}, \Sigma\pi_2\rangle \le (B\mu_X \times \kappa_X) \circ \langle (\tau_2)_{\Sigma^* X}, \Sigma\pi_2\rangle$$

by monotonicity of $B\mu$ and assumption. Now using the definition of $(\tau_1^\dagger)_X$, it easily follows by induction on terms in $\Sigma^*(BX \times X)$ that $(\tau_1^\dagger)_X \le (\tau_2^\dagger)_X$, and thus $\pi_1 \circ (\tau_1^\dagger)_X \le \pi_1 \circ (\tau_2^\dagger)_X$. $\square$

Because φ is monotone, it has a least fixed point, which we denote by $\mathsf{lfp}(\varphi)$. Further, since φ preserves monotonicity we obtain monotonicity of $\mathsf{lfp}(\varphi)$ by transfinite induction (the base case and limit steps are rather easy). The proof technique of transfinite induction, which we also use several times below, is justified by the fact that the 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 (see, e.g., [San12a]).

Corollary 6.2.7. *The abstract GSOS specification* $\mathsf{lfp}(\varphi)$ *is monotone.*

Informally, $\mathsf{lfp}(\varphi)$ is the specification consisting of rules from ρ and Δ. 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 6.1.7), 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 6.2.8. *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 $\psi(f) = f$.

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

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 $B\mu_\emptyset$ preserves joins. For the right-hand part, we have

$$
\begin{array}{lll}
 & \bigvee_{d\in\Delta} B\mu_\emptyset \circ \pi_1 \circ (\mathsf{lfp}(\varphi))^\dagger_{\Sigma^*\emptyset} \circ d_{B\Sigma^*\emptyset\times\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle & \\
= & \bigvee_{d\in\Delta} \pi_1 \circ B\mu_\emptyset \times \mu_\emptyset \circ (\mathsf{lfp}(\varphi))^\dagger_{\Sigma^*\emptyset} \circ \Sigma^*\langle f, \mathsf{id}\rangle \circ d_{\Sigma^*\emptyset} & \text{naturality of } d,\ \pi_1 \\
= & \bigvee_{d\in\Delta} \pi_1 \circ \langle f, \mathsf{id}\rangle \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset} & (\Sigma\emptyset^*, \mu_\emptyset, \langle f, \mathsf{id}\rangle) \text{ is a} \\
= & \bigvee_{d\in\Delta} f \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset} & (\mathsf{lfp}(\varphi))^\dagger\text{-bialg.}
\end{array}
$$

Thus $f\circ\kappa_\emptyset = B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{d\in\Delta} f \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset} = \psi(f) \circ \kappa_\emptyset$ and consequently $\psi(f) = f$, since $\kappa_\emptyset$ 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 6.2.8, we then have $M(\mathsf{lfp}(\varphi)) = \mathsf{lfp}(\psi)$ (Theorem 6.2.14). The main step is that any fixed point of ψ is "closed under ρ", i.e., that in such a model, all the behaviour that we can derive by the specification is already there. This result is the contents of Corollary 6.2.13 below; it follows by transfinite induction from Lemma 6.2.11 and 6.2.12. But first, we need a few technical tools (Lemma 6.2.9 and 6.2.10). Recall from Section 3.4 that a Σ-algebra $\alpha\colon \Sigma X \to X$ induces an algebra $\widehat{\alpha}\colon \Sigma^* X \to X$ for the free monad. This construction preserves algebra morphisms. We prove a lax version of this fact.

Lemma 6.2.9. *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* $f\colon X \to Y$*:*

$$
\begin{tikzcd}
\Sigma X \arrow[r, "\Sigma f"] \arrow[d, "\alpha"'] \arrow[dr, phantom, "\geq"] & \Sigma Y \arrow[d, "\beta"] \\
X \arrow[r, "f"'] & Y
\end{tikzcd}
\quad \textit{implies} \quad
\begin{tikzcd}
\Sigma^* X \arrow[r, "\Sigma^* f"] \arrow[d, "\widehat{\alpha}"'] \arrow[dr, phantom, "\geq"] & \Sigma^* Y \arrow[d, "\widehat{\beta}"] \\
X \arrow[r, "f"'] & Y
\end{tikzcd}
$$

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

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

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

$$\begin{array}{ll}
\widehat{\beta} \circ \Sigma^* f \circ \kappa_X(\sigma(t_1, \ldots, t_n)) & \\
= \widehat{\beta} \circ \kappa_Y \circ \Sigma\Sigma^* f(\sigma(t_1, \ldots, t_n)) & \text{naturality } \kappa \\
= \widehat{\beta} \circ \kappa_Y(\sigma(\Sigma^* f(t_1), \ldots, \Sigma^* f(t_n))) & \text{definition } \Sigma \\
= \beta \circ \Sigma\widehat{\beta}(\sigma(\Sigma^* f(t_1), \ldots, \Sigma^* f(t_n))) & \text{definition } \widehat{\beta} \\
= \beta(\sigma(\widehat{\beta} \circ \Sigma^* f(t_1), \ldots, \widehat{\beta} \circ \Sigma^* f(t_n))) & \text{definition } \Sigma \\
\leq \beta(\sigma(f \circ \widehat{\alpha}(t_1), \ldots, f \circ \widehat{\alpha}(t_n))) & \text{ind. hypothesis, monotonicity } \beta \\
= \beta \circ \Sigma f \circ \Sigma\widehat{\alpha}(\sigma(t_1, \ldots, t_n)) & \text{definition } \Sigma \\
\leq f \circ \alpha \circ \Sigma\widehat{\alpha}(\sigma(t_1, \ldots, t_n)) & \text{assumption} \\
= f \circ \widehat{\alpha} \circ \kappa_X(\sigma(t_1, \ldots, t_n)) & \text{definition } \widehat{\alpha}
\end{array}$$

which concludes the induction step. ☐

We instantiate the above lemma to the definition of $\tau^\dagger$.

Lemma 6.2.10. *Let τ be a monotone abstract GSOS specification of Σ over B. Then for any $f\colon \Sigma^*\emptyset \to B\Sigma^*\emptyset$:*

$$\begin{tikzcd}
\Sigma\Sigma^*\emptyset \arrow[r, "\Sigma\langle f, \mathsf{id}\rangle"] \arrow[dd, "\kappa_\emptyset"'] & \Sigma(B\Sigma^*\emptyset \times \Sigma^*\emptyset) \arrow[d, "\tau_{\Sigma^*\emptyset}"] \\
\geq & B\Sigma^*\Sigma^*\emptyset \arrow[d, "B\mu_\emptyset"] \\
\Sigma^*\emptyset \arrow[r, "f"'] & B\Sigma^*\emptyset
\end{tikzcd}
\quad \textit{implies} \quad
\begin{tikzcd}
\Sigma^*\Sigma^*\emptyset \arrow[r, "\Sigma^*\langle f, \mathsf{id}\rangle"] \arrow[dd, "\mu_\emptyset"'] & \Sigma^*(B\Sigma^*\emptyset \times \Sigma^*\emptyset) \arrow[d, "\tau^\dagger_{\Sigma^*\emptyset}"] \\
\geq & B\Sigma^*\Sigma^*\emptyset \times \Sigma^*\Sigma^*\emptyset \arrow[d, "B\mu_\emptyset \times \mu_\emptyset"] \\
\Sigma^*\emptyset \arrow[r, "\langle f, \mathsf{id}\rangle"'] & B\Sigma^*\emptyset \times \Sigma^*\emptyset
\end{tikzcd}$$

Proof. From the assumption it follows that

$$(B\mu_\emptyset \times \kappa_\emptyset) \circ \langle \tau_{\Sigma^*\emptyset}, \Sigma\pi_2\rangle \circ \Sigma\langle f, \mathsf{id}\rangle \leq \langle f, \mathsf{id}\rangle \circ \kappa_\emptyset\,.$$

Let $\beta = (B\mu_\emptyset \times \kappa_\emptyset) \circ \langle \tau_{\Sigma^*\emptyset}, \Sigma\pi_2\rangle$, then by Lemma 6.2.9 we get

$$\begin{tikzcd}
\Sigma^*\Sigma^*\emptyset \arrow[r, "\Sigma^*\langle f, \mathsf{id}\rangle"] \arrow[d, "\mu_\emptyset"'] \arrow[dr, phantom, "\geq"] & \Sigma^*(B\Sigma^*\emptyset \times \Sigma^*\emptyset) \arrow[d, "\widehat{\beta}"] \\
\Sigma^*\emptyset \arrow[r, "\langle f, \mathsf{id}\rangle"'] & B\Sigma^*\emptyset \times \Sigma^*\emptyset
\end{tikzcd}$$

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

To this end, consider the following diagram:

$$\begin{tikzcd}
\Sigma\Sigma^*(B\Sigma^*\emptyset \times \Sigma^*\emptyset) \arrow[r, "\Sigma\tau^\dagger_{\Sigma^*\emptyset}"] \arrow[dd, "\kappa_{B\Sigma^*\emptyset\times\Sigma^*\emptyset}"'] & \Sigma(B\Sigma^*\Sigma^*\emptyset \times \Sigma^*\Sigma^*\emptyset) \arrow[r, "\Sigma(B\mu_\emptyset\times\mu_\emptyset)"] \arrow[d, "\langle\tau_{\Sigma^*\Sigma^*\emptyset},\Sigma\pi_2\rangle"] & \Sigma(B\Sigma^*\emptyset \times \Sigma^*\emptyset) \arrow[d, "\langle\tau_{\Sigma^*\emptyset},\Sigma\pi_2\rangle"] \\
 & B\Sigma^*\Sigma^*\Sigma^*\emptyset \times \Sigma\Sigma^*\Sigma^*\emptyset \arrow[r, "B\Sigma^*\mu_\emptyset\times\Sigma\mu_\emptyset"] \arrow[d, "B\mu_{\Sigma^*\emptyset}\times\kappa_{\Sigma^*\emptyset}"] & B\Sigma^*\Sigma^*\emptyset \times \Sigma\Sigma^*\emptyset \arrow[d, "B\mu_\emptyset\times\kappa_\emptyset"] \\
\Sigma^*(B\Sigma^*\emptyset \times \Sigma^*\emptyset) \arrow[r, "\tau^\dagger_{\Sigma^*\emptyset}"] & B\Sigma^*\Sigma^*\emptyset \times \Sigma^*\Sigma^*\emptyset \arrow[r, "B\mu_\emptyset\times\mu_\emptyset"] & B\Sigma^*\emptyset \times \Sigma^*\emptyset \\
B\Sigma^*\emptyset \times \Sigma^*\emptyset \arrow[u, "\eta_{B\Sigma^*\emptyset\times\Sigma^*\emptyset}"] \arrow[ur, "B\eta_{\Sigma^*\emptyset}\times\eta_{\Sigma^*\emptyset}"'] & &
\end{tikzcd}$$

The upper right rectangle commutes by naturality, the lower right rectangle commutes by the multiplication law of the monad and since $\mu_\emptyset = \widehat{\kappa_\emptyset}$. The left square and triangle commute by definition of $\tau^\dagger$ (Equation (3.15) in Section 3.5.2). Thus $(B\mu_\emptyset \times \mu_\emptyset) \circ \tau^\dagger_{\Sigma^*\emptyset}$ 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 $\widehat{\beta} = B\mu_\emptyset \times \mu_\emptyset \circ \tau^\dagger_{\Sigma^*\emptyset}$. □

Lemma 6.2.11. *Let τ be a specification, and $f \in \mathbb{M}$ a fixed point of ψ. If $B\mu_\emptyset \circ \tau_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq f \circ \kappa_\emptyset$ then $B\mu_\emptyset \circ \varphi(\tau)_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq f \circ \kappa_\emptyset$.*

Proof.

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

The first equality holds by definition of φ, the second by definition of the join of specifications, the third since $B\mu_\emptyset$ is join-preserving, and the fourth equality by naturality of d and π_1. The inequality holds by assumption and Lemma 6.2.10. The last equality holds by definition of ψ. $\square$

Lemma 6.2.12. *Let $f \in \mathbb{M}$ such that $\psi(f) = f$, and suppose we have a family $\{\tau_i\}_{i\in I}$ of specifications, for some index set I. If $B\mu_\emptyset \circ (\tau_i)_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq f \circ \kappa_\emptyset$ for all $i \in I$, then $B\mu_\emptyset \circ (\bigvee_{i\in I} \tau_i)_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq f \circ \kappa_\emptyset$.*

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

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

and the result now follows by the assumption that $B\mu_\emptyset \circ (\tau_i)_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \leq f \circ \kappa_\emptyset$ for each i. $\square$

Corollary 6.2.13. *For any $f \in \mathbb{M}$: if $\psi(f) = f$ then*

$$\begin{tikzcd}
\Sigma\Sigma^* X \arrow[r, "\Sigma\langle f,\mathsf{id}\rangle"] \arrow[dd, "\kappa_\emptyset"'] & \Sigma(B\Sigma^*\emptyset \times \Sigma^*\emptyset) \arrow[d, "\mathsf{lfp}(\varphi)_{\Sigma^*\emptyset}"] \\
\arrow[r, phantom, "\geq"] & B\Sigma^*\Sigma^*\emptyset \arrow[d, "B\mu_\emptyset"] \\
\Sigma^* X \arrow[r, "f"'] & B\Sigma^*\emptyset
\end{tikzcd}$$

Proof. By transfinite induction. For the base case we have $B\mu_\emptyset \circ \perp \circ \Sigma\langle f, \mathsf{id}\rangle = \perp \leq f \circ \kappa_\emptyset$. The successor step is given by Lemma 6.2.11 and the limit step by Lemma 6.2.12. $\square$

This allows to prove the main result of this chapter.

Theorem 6.2.14. *The interpretation of ρ and Δ coincides with the operational model of the abstract GSOS specification $\mathsf{lfp}(\varphi)$, i.e., $M(\mathsf{lfp}(\varphi)) = \mathsf{lfp}(\psi)$.*

Proof. By Lemma 6.2.8, $M(\mathsf{lfp}(\varphi))$ 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 $t_1, \ldots, t_n \in \Sigma^*\emptyset$ 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 $n = 0$). Then

$$\begin{aligned}
M(\mathsf{lfp}(\varphi))(\sigma(t_1, \ldots, t_n)) &= B\mu_\emptyset \circ (\mathsf{lfp}(\varphi))_{\Sigma^*\emptyset} \circ \Sigma\langle M(\mathsf{lfp}(\varphi)), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) \\
&\leq B\mu_\emptyset \circ (\mathsf{lfp}(\varphi))_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) \\
&\leq f(\sigma(t_1, \ldots, t_n))
\end{aligned}$$

where the first inequality holds by assumption and monotonicity of $B\mu_\emptyset$ and $\mathsf{lfp}(\varphi)$ (Corollary 6.2.7) and the second by Corollary 6.2.13. $\square$

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

Corollary 6.2.15. *Bisimilarity is a congruence on the interpretation* $\mathsf{lfp}(\psi)$ *of* ρ *and* Δ*, and bisimulation up to context is compatible (i.e., the contextual closure is* $\mathsf{b}_{\mathsf{lfp}(\psi)}$*-compatible).*

Example 6.2.16. The parallel composition can be given by a positive GSOS specification, and Equation (6.1) of the introduction contains a rule for the replication operator. Thus, by the above Corollary, bisimilarity is a congruence on the operational model of CCS with replication, and bisimulation up to context is compatible; this is known (see, e.g., [San12a]), but here we obtain it directly from the format and the above results.

Example 6.2.17. We revisit the general process algebra with transition costs (GPA) (see Example 4.5.5, [BK01]). We consider basic GPA processes with procedures, 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 the operators of basic GPA processes on the complete monoid $\mathbb{R}^+ \cup \{\infty\}$ (with supremum) is similar to the semantics in Example 4.5.5. The semantics corresponds to a GSOS specification; see [Kli11] for details. This specification is monotone. The (recursive) procedures can now be interpreted by assignment rules: 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 6.2.14, bisimilarity is a congruence on the interpretation.

6.3 Structural congruences

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

Definition 6.3.1. Let $E \subseteq \Sigma^* V \times \Sigma^* V$ be a set of equations. The *congruence closure* $\equiv_E$ of E is the least relation on $\Sigma^* \emptyset$ satisfying the following rules:

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

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

where $s^\sharp \colon \Sigma^* V \to \Sigma^* \emptyset$ is the inductive extension of s to terms (Section 3.4).

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

$$\frac{t \equiv_E u \qquad u \xrightarrow{a} u' \qquad u' \equiv_E v}{t \xrightarrow{a} v} \tag{6.7}$$

Informally, this rule states that we can use the specification to derive transitions modulo the congruence generated by the equations. In fact, removing the part $u' \equiv_E v$ from the premise (and writing u' instead of v in the conclusion) does not affect the behaviour, modulo bisimilarity [MR05]. See [MR05] for details on the interpretation of structural congruences in the context of transition systems.

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

Definition 6.3.2. Let $\theta \colon \mathbb{M} \to \mathbb{M}$ be the (unique) function such that

$$\theta(f) \circ \kappa_\emptyset = B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle f, \mathsf{id}\rangle \vee \bigvee_{r \in R} f \circ r \circ q \circ \kappa_\emptyset \, .$$

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 6.3.3. *The function $\theta \colon \mathbb{M} \to \mathbb{M}$ is monotone.*

Proof. Similar to the proof of Lemma 6.1.8. □

Definition 6.3.4. The *interpretation* of ρ and E is the least (ρ, E)-model, i.e., $\mathsf{lfp}(\theta)$.

Example 6.3.5. Consider the specification of the parallel composition $x|y$ as given in (6.2) in the introduction of this chapter, i.e., by a single rule and commutativity. In the interpretation, if $t \xrightarrow{a} t'$ then $t|u \xrightarrow{a} t'|u$, simply by the SOS rule. But also $u|t \xrightarrow{a} t'|u$, since $t|u \equiv_E u|t$. Concerning the definition of the replication operator by the equation $!x = !x|x$, for a term t the interpretation contains the least set of transitions from $!t$ which satisfy the equation, as desired.

In general, bisimilarity is not a congruence when equations are added. For convenience we recall a counterexample on transition systems [MR05].

Example 6.3.6. Consider rules $\overline{p \xrightarrow{a} p}$ and $\overline{q \xrightarrow{a} p}$ 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 its arguments. To rule out such assignments, a restricted format of equations was introduced in [MR05], called cfsc. The main result of [MR05] is that for any specification in the tyft format combined with cfsc equations, bisimilarity is a congruence.

Definition 6.3.7. A set of equations $E \subseteq \Sigma^* V \times \Sigma^* V$ is in cfsc 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 $\rho_X(\sigma(u_1, \ldots, u_n)) = \perp$ for any set X and any $u_1, \ldots, u_n \in BX \times X$.

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 (6.2). 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 6.2.17 can be modelled by defining equations. Associativity of $|$ is neither a σx-equation nor a defining one. We refer to [MR05] 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 can not assign behaviour to complex terms, disallowing a situation such as in Example 6.3.6.

We proceed to show that the interpretation of an abstract GSOS specification ρ and a set of equations E in cfsc equals the operational model of a certain other specification, up to bisimilarity (Definition 4.4.10). 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.

Definition 6.3.8. A set of equations E in cfsc 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 and 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

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

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

Remark 6.3.9. In [MR05], σx-equations are a bit more liberal in that they do not require the arities of σ_1 and σ_2 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 can 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 $t \in \Sigma^*\emptyset$ mapping $\sigma_1(x)$ to $\sigma_2(x, t)$. We work with the simpler format above for technical convenience.

We prove that the encoding of equations as assignment rules is correct with respect to the interpretation of the equations (Theorem 6.3.13). 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 6.3.10. *Let E be a set of equations in cfsc format w.r.t. ρ, and let ψ be as in Definition 6.1.7 for (ρ, Δ^E). Then for any defining equation $\sigma(x_1, \ldots, x_n) = t$ and any $t_1, \ldots, t_n \in \Sigma^*\emptyset$: $\mathsf{lfp}(\psi) \circ \kappa_\emptyset(\sigma(t_1, \ldots, t_n)) \leq \mathsf{lfp}(\psi) \circ \mu_\emptyset(t[x_1 := t_1, \ldots, x_n := t_n])$.*

Proof. Given a defining equation, let $d \in \Delta^E$ be the natural transformation that encodes it (see Definition 6.3.8(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 $t_1, \ldots, t_n \in \Sigma^*\emptyset$ we have

$$g \circ \kappa_\emptyset(\sigma(t_1, \ldots, t_n)) \leq \mathsf{lfp}(\psi) \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset}(\sigma(t_1, \ldots, t_n)) \,. \tag{6.8}$$

The base case is when $g = \bot$, which is trivial. Now suppose that (6.8) holds for some $g \leq \mathsf{lfp}(\psi)$. Then

$$\psi(g) \circ \kappa_\emptyset(\sigma(t_1, \ldots, t_n)) = (B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle g, \mathsf{id}\rangle \vee \bigvee_{d' \in \Delta^E} g \circ \mu_\emptyset \circ d'_{\Sigma^*\emptyset})(\sigma(t_1, \ldots, t_n)) \,.$$

But since the equations are in cfsc format, we have

$$B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle g, \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) = \bot \,. \tag{6.9}$$

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

$$g \circ \mu_\emptyset \circ d'_{\Sigma^*\emptyset}(\sigma(t_1, \ldots, t_n)) = g \circ \kappa_\emptyset(\sigma(t_1, \ldots, t_n))$$

which is below $\mathsf{lfp}(\psi) \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset}(\sigma(t_1, \ldots, t_n))$ by the induction hypothesis (6.8). Together with the assumption that $g \leq \mathsf{lfp}(\psi)$ this implies

$$\bigvee_{d' \in \Delta^E} g \circ \mu_\emptyset \circ d'_{\Sigma^*\emptyset}(\sigma(t_1, \ldots, t_n)) \leq \mathsf{lfp}(\psi) \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset}(\sigma(t_1, \ldots, t_n)) \,.$$

By the above and (6.9), we may conclude

$$\psi(g) \circ \kappa_\emptyset(\sigma(t_1, \ldots, t_n)) \leq \mathsf{lfp}(\psi) \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset}(\sigma(t_1, \ldots, t_n))$$

as desired. This concludes the successor step; the limit step is again trivial (i.e., if we assume that (6.8) 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.

Lemma 6.3.11. *Let E and ψ be as above. If $t \equiv_E u$ then $Bq \circ (\mathsf{lfp}(\psi))(t) = Bq \circ (\mathsf{lfp}(\psi))(u)$, 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 $Bq \circ (\mathsf{lfp}(\psi))(t) = Bq \circ (\mathsf{lfp}(\psi))(u)$ 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 $\mu_\emptyset \circ d_{\Sigma^*\emptyset}(\sigma_1(t_1, \ldots, t_n)) = \sigma_2(u_1, \ldots, u_n)$, and by definition of $\mathsf{lfp}(\psi)$ we have $\mathsf{lfp}(\psi) \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset} \leq \mathsf{lfp}(\psi)$; 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, there is another assignment rule d', and thus $(\mathsf{lfp}(\psi))(\sigma_1(t_1, \ldots, t_n)) \leq (\mathsf{lfp}(\psi))(\sigma_2(u_1, \ldots, u_n))$.

For a defining equation $\sigma(t_1, \ldots, t_n) \equiv_E t$ we have a natural transformation in d such that $\mu_\emptyset \circ d_{\Sigma^*\emptyset}(\sigma(t_1, \ldots, t_n)) = t$. Thus $(\mathsf{lfp}(\psi))(t) = (\mathsf{lfp}(\psi)) \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset}(\sigma(t_1, \ldots, t_n)) \leq (\mathsf{lfp}(\psi))(\sigma(t_1, \ldots, t_n))$. The other way around follows by Lemma 6.3.10. 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 $t_i \equiv u_i$ and $Bq \circ (\mathsf{lfp}(\psi))(t_i) = Bq \circ (\mathsf{lfp}(\psi))(u_i)$ for all $i \leq n$, and σ is an operator of arity n. Notice that this implies

$$\langle Bq \circ \mathsf{lfp}(\psi), q\rangle(t_i) = \langle Bq \circ \mathsf{lfp}(\psi), q\rangle(u_i) \qquad \text{for all } i \leq n \tag{6.10}$$

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

$$\begin{array}{ll}
Bq \circ (\mathsf{lfp}(\psi))(\sigma(t_1, \ldots, t_n)) & \\
= Bq \circ B\mu_\emptyset \circ (\mathsf{lfp}(\varphi))_{\Sigma^*\emptyset} \circ \Sigma\langle \mathsf{lfp}(\psi), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) & \text{Theorem 6.2.14} \\
= B\mu' \circ B\Sigma^* q \circ (\mathsf{lfp}(\varphi))_{\Sigma^*\emptyset} \circ \Sigma\langle \mathsf{lfp}(\psi), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) & q \text{ alg. morphism} \\
= B\mu' \circ (\mathsf{lfp}(\varphi))_{\Sigma^*\emptyset} \circ \Sigma(Bq \times q) \circ \Sigma\langle \mathsf{lfp}(\psi), \mathsf{id}\rangle(\sigma(t_1, \ldots, t_n)) & \text{naturality} \\
= B\mu' \circ (\mathsf{lfp}(\varphi))_{\Sigma^*\emptyset} \circ \Sigma\langle Bq \circ \mathsf{lfp}(\psi), q\rangle(\sigma(t_1, \ldots, t_n)) & \text{functoriality} \\
= B\mu' \circ (\mathsf{lfp}(\varphi))_{\Sigma^*\emptyset} \circ \Sigma\langle Bq \circ \mathsf{lfp}(\psi), q\rangle(\sigma(u_1, \ldots, u_n)) & \text{ind. hypothesis} \\
= Bq \circ B\mu_\emptyset \circ (\mathsf{lfp}(\varphi))_{\Sigma^*\emptyset} \circ \Sigma\langle \mathsf{lfp}(\psi), \mathsf{id}\rangle(\sigma(u_1, \ldots, u_n)) & \\
= Bq \circ (\mathsf{lfp}(\psi))(\sigma(u_1, \ldots, u_n)) &
\end{array}$$

Notice that we used the fact that the quotient map q is an algebra morphism into some Σ^*-algebra μ'. It is worthwhile to note that we need to reason up to $\equiv_E$ to get (6.10). 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 necessary equal. □

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

Lemma 6.3.12. *Let ψ and q be as above. Then $Bq \circ (\mathsf{lfp}(\theta)) = Bq \circ (\mathsf{lfp}(\psi))$.*

Proof. We first prove that $\psi(\mathsf{lfp}(\theta)) \leq \mathsf{lfp}(\theta)$. The interesting part is to show that $\mathsf{lfp}(\theta) \circ \mu_\emptyset \circ d_{\Sigma^*\emptyset} \leq \mathsf{lfp}(\theta) \circ \kappa_\emptyset$ for any $d \in \Delta^E$, given that $\bigvee_{r\in R} \mathsf{lfp}(\theta) \circ r \circ q \circ \kappa_\emptyset \leq \mathsf{lfp}(\theta) \circ \kappa_\emptyset$ (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 $\psi(\mathsf{lfp}(\theta)) \leq \mathsf{lfp}(\theta)$, and since $\mathsf{lfp}(\psi)$ is the least pre-fixed point of ψ we have $\mathsf{lfp}(\psi) \leq \mathsf{lfp}(\theta)$. Hence $Bq \circ \mathsf{lfp}(\psi) \leq Bq \circ \mathsf{lfp}(\theta)$.

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

$$\begin{aligned} Bq \circ \theta(h) \circ \kappa_\emptyset &= Bq \circ (B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle h, \mathsf{id}\rangle \vee \bigvee_{r\in R} h \circ r \circ q \circ \kappa_\emptyset) \\ &= Bq \circ B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle h, \mathsf{id}\rangle \vee \bigvee_{r\in R} Bq \circ h \circ r \circ q \circ \kappa_\emptyset \end{aligned}$$

Now

$$\begin{aligned} Bq \circ B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle h, \mathsf{id}\rangle &= B\mu' \circ B\Sigma^* q \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle h, \mathsf{id}\rangle \\ &= B\mu' \circ \rho_{\Sigma^*\emptyset} \circ \Sigma(Bq \times q) \circ \Sigma\langle h, \mathsf{id}\rangle \\ &\leq B\mu' \circ \rho_{\Sigma^*\emptyset} \circ \Sigma(Bq \times q) \circ \Sigma\langle \mathsf{lfp}(\psi), \mathsf{id}\rangle \\ &= Bq \circ B\mu_\emptyset \circ \rho_{\Sigma^*\emptyset} \circ \Sigma\langle \mathsf{lfp}(\psi), \mathsf{id}\rangle \\ &\leq Bq \circ \mathsf{lfp}(\psi) \circ \kappa_\emptyset \end{aligned}$$

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

$$\bigvee_{r\in R} Bq \circ h \circ r \circ q \circ \kappa_\emptyset \leq \bigvee_{r\in R} Bq \circ \mathsf{lfp}(\psi) \circ r \circ q \circ \kappa_\emptyset = Bq \circ \mathsf{lfp}(\psi) \circ \kappa_\emptyset$$

by assumption and Lemma 6.3.11. Thus $Bq \circ \theta(h) \leq Bq \circ \mathsf{lfp}(\psi)$ as desired. □

This implies that $\mathsf{lfp}(\theta)$ and $\mathsf{lfp}(\psi)$ are *behaviourally equivalent* up to $\equiv_E$. Recall that behavioural equivalence coincides with bisimilarity whenever the functor B preserves weak pullbacks (Lemma 3.1.6). Under this assumption one can prove that $\mathsf{lfp}(\theta)$ is equal to $\mathsf{lfp}(\psi)$ up to bisimilarity, and by Theorem 6.2.14 we then obtain our main result of this section.

Theorem 6.3.13. *Suppose E is a set of equations which is in cfsc format w.r.t. ρ, and suppose the behaviour functor B preserves weak pullbacks. Then the interpretation $\mathsf{lfp}(\theta)$ of ρ and E equals the operational model of a certain abstract GSOS specification, up to bisimilarity (Definition 4.4.10). Bisimilarity is a congruence, and* $\mathsf{bis} \circ \mathsf{ctx} \circ \mathsf{bis}$ *is* $\mathsf{b}_{\mathsf{lfp}(\theta)}$*-compatible.*

Proof. Using the universal property of the coequalizer $q\colon \Sigma^*\emptyset \to (\Sigma^*\emptyset)/\equiv_E$, by Lemma 6.3.11 we obtain a unique coalgebra structure on $(\Sigma^*\emptyset)/\equiv_E$ turning q into a homomorphism:

$$\begin{tikzcd}
\equiv_E \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\pi_2"'] & \Sigma^*\emptyset \arrow[r, "q"] \arrow[d, "\mathsf{lfp}(\psi)"'] & (\Sigma^*\emptyset)/\equiv_E \arrow[d, dashed] \\
& B\Sigma^*\emptyset \arrow[r, "Bq"] & B(\Sigma^*\emptyset)/\equiv_E
\end{tikzcd}$$

Further, by Lemma 6.3.12, q is also a homomorphism from $\mathsf{lfp}(\theta)$ into the same coalgebra. Now the pullback (in Set) of q along itself is simply $\equiv_E$, and since B preserves weak pullbacks, $\equiv_E$ is a bisimulation between $\mathsf{lfp}(\psi)$ and $\mathsf{lfp}(\theta)$ [Rut00, Theorem 4.3]. Thus, in particular, $\mathsf{lfp}(\psi)$ and $\mathsf{lfp}(\theta)$ are equal up to bisimilarity, since $\equiv_E$ is reflexive.

By Theorem 6.2.14, bisimilarity is a congruence on $\mathsf{lfp}(\psi)$. Since $\mathsf{lfp}(\psi)$ and $\mathsf{lfp}(\theta)$ are equal up to bisimilarity, it follows from Lemma 4.4.11 that bisimilarity is a congruence on $\mathsf{lfp}(\theta)$. Finally, again by Theorem 6.2.14, ctx is $\mathsf{b}_{\mathsf{lfp}(\psi)}$-compatible. Thus, by Lemma 4.4.12, $\mathsf{bis} \circ \mathsf{ctx} \circ \mathsf{bis}$ is $\mathsf{b}_{\mathsf{lfp}(\theta)}$-compatible. □

6.4 Discussion and related work

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. 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 [MR05].

The main work in the literature that treats the meta-theory of rule formats with structural congruences [MR05] 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 [MR05] is tyft/tyxt[2], which is more expressive than positive GSOS since it allows lookahead in the premises. However, while [MR05] proves congruence of bisimilarity this does not imply the compatibility (or even soundness) of bisimulation up to

[2] In [MR05], 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.

context [PS12], 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 [Plo01]. Klin [Kli04] 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 [Kli04]. Further, in [Kli04] each operator is either specified by an equation or by operational rules, disallowing a specification such as that of the parallel composition in equation (6.2).

In [LPW04], various constructions on distributive laws are presented. Example 32 of that paper discusses the definition of the parallel composition as in (6.2) above, but a general theory for structural congruence is missing. Distributive laws are applied in [Jac06b] to find solutions of guarded recursive equations. Further, in [MMS13] 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 [BM02, CHM02], it is shown how to lift calculi with structural axioms to coalgebraic models, but under the assumption that the equations already hold.

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., [Kli11]). While in general the combined use of cofree comonads and free monads in specifications is known to be problematic [KN14], 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 the next chapter.

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