Abstract Congruence Criteria for Weak Bisimilarity

Stelios Tsampas
KU Leuven, Belgium

Christian Williams
University of California, Riverside, CA, USA

Andreas Nuyts
Vrije Universiteit Brussel, Belgium

Dominique Devriese
Vrije Universiteit Brussel, Belgium

Frank Piessens
KU Leuven, Belgium

Abstract

We introduce three general compositionality criteria over operational semantics and prove that, when all three are satisfied together, they guarantee weak bisimulation being a congruence. Our work is founded upon Turi and Plotkin's mathematical operational semantics and the coalgebraic approach to weak bisimulation by Brengos. We demonstrate each criterion with various examples of success and failure and establish a formal connection with the simply WB cool rule format of Bloom and van Glabbeek. In addition, we show that the three criteria induce lax models in the sense of Bonchi et al.

2012 ACM Subject Classification Theory of computation → Categorical semantics

Keywords and phrases Structural Operational Semantics, distributive laws, weak bisimilarity

Digital Object Identifier 10.4230/LIPIcs.MFCS.2021.88

Acknowledgements This research was partially funded by the Research Fund KU Leuven. Andreas Nuyts was supported by a grant of the Research Foundation – Flanders (FWO).

1 Introduction

The problem of *full abstraction* for programming language semantics [21, 13], i.e. the perfect agreement between a denotational and an operational specification, has been both significant and enduring. It requires that the denotational semantics, which bestows each program with a denotation, a meaning, is sufficiently coarse that it does not distinguish terms behaving the same operationally. At the same time, the denotational semantics must remain a *congruence* [36], to make the semantics *compositional*: the denotation of a composite term is fully determined by the denotations of its subterms irrespective of their internal structure.

From an operational point of view, the choice of behavioral equivalence is generally open. Bisimilarity, trace equivalence, weak bisimilarity etc. are all potentially applicable. However, bisimilarity is often too strong for practical purposes. For instance, labelled transition systems [39] (LTS) may perform invisible steps which are not ignored by bisimilarity as opposed to *weak* bisimilarity [18, 19]. This is taken a step further in programming language semantics, where a natural definition of program equivalence is that of the *largest adequate* (w.r.t. observing termination) *congruence* relation [25]. This relation is also known as *contextual equivalence*, the crown jewel of program equivalences, and can be reformulated in a more explicit manner as *Morris-style* contextual equivalence [20, 13, 24], i.e. indistinguishability under all program contexts.

Despite being the subject of vigorous research, proving that coarser behavioral equivalences are congruences remains a hard problem. Weak bisimilarity in particular stands out as a popular equivalence that has seen widespread usage in the literature yet has been proven hard to reason with [30, 31, 3]. To that end, various powerful methods exist for proving congruence-closedness of bisimilarity, like Howe's method [15, 16] and *logical relations* [10, 22, 23], yet the machinery involved is complicated and non-trivial to execute correctly. In addition, there have been the so-called *cool congruence formats* for weak bisimulation introduced by Bloom [4] and van Glabbeek [37] that ensure weak bisimilarity being a congruence, but only if the semantics adhere to the rule formats.

In this work, we propose three general *compositionality criteria* over operational specifications that, when all three are satisfied, guarantee weak bisimilarity being a congruence. The foundation of our approach is categorical: on the one hand, the framework of *mathematical operational semantics* introduced by Turi and Plotkin [35] acts as an ideal abstract setting to explore programming language semantics. On the other hand, the coalgebraic approach introduced by Brengos [8, 9] describes a seamless way to define weak bisimulation in categories of coalgebras, should the underlying behavior be monadic and equipped with an order structure. With that in mind, the three criteria, which we name *continuity*, *unitality* and *observability*, essentially characterize how the semantics interact with the order structure of the behavior.

Related work

As mentioned earlier, Bloom and van Glabbeek have introduced the cool congruence formats; weak bisimulation for any system given in these formats is guaranteed to be a congruence. In this work we are also able to establish a formal connection with the *simply WB cool* format for LTSs (Theorem 3.16), specifically that any system given in the simply WB cool rule format automatically satisfies the three criteria. In that sense, our three criteria form a broader, less restrictive approach on the same problem (extended beyond LTSs). Furthermore, the many examples provided by van Glabbeek [37] help explain this connection and at the same time act as excellent hands-on case studies for our three criteria.

Apart from the aforementioned work of Brengos, various approaches at the hard problem of coalgebraic weak bisimulation have been proposed [27, 29, 12, 5]. Of particular interest is the work of Bonchi et al. [6] on up-to techniques [7] for weak bisimulations in the context of mathematical operational semantics. The main theoretical device in their work is that of a *lax model*, a relaxation of the notion of a *bialgebra*. We relate lax models with our work by showing that specifications satisfying the three criteria induce lax models (Theorem 3.21) and argue that, as a formal method, our criteria are significantly easier to prove.

Paper outline

In Section 2.1 we introduce Turi and Plotkin's mathematical operational semantics [35] as well as the two "running" example systems used throughout the paper. In Section 2.2 we present the work of Brengos on weak bisimulation [8, 9] and show how it applies to our two examples. We expand on our contribution in Section 3, where we introduce the three criteria and put them to the test against our main examples as well as examples from van Glabbeek [37]. We subsequently formalize the connection of our three criteria with the simply WB cool rule format (Theorem 3.16) and then move on to present our main theorem (Theorem 3.18). Finally, in Section 3.3 we show how our three criteria induce lax models in the sense of Bonchi et al. [6].

2 Preliminaries

2.1 Mathematical Operational Semantics

We first summarize the basic framework of Turi and Plotkin's *mathematical operational semantics* [35]. The idea is that operational semantics correspond to *distributive laws* of varying complexity on a base category $\mathbb{C}$. For our work we need only consider the most important form of distributive laws, that of *GSOS laws* [35], as they are in an 1-1 correspondence with the historically significant *GSOS* rule format [26].

▶ **Definition 2.1.** *Let endofunctors $\Sigma, B : \mathbb{C} \to \mathbb{C}$ on a cartesian category $\mathbb{C}$. A GSOS law of Σ over B is a natural transformation $\rho : \Sigma(\mathrm{Id} \times B) \Longrightarrow B\Sigma^*$, where Σ^* is the free monad over Σ.*

Endofunctors Σ and B are understood as the syntax and behavior (resp.) of a system whereas ρ represents the semantics. Over the course of the paper we will also be using an alternative representation of GSOS laws given by the following correspondence.

▶ **Proposition 2.2.** *GSOS laws $\rho : \Sigma(\mathrm{Id} \times B) \Longrightarrow B\Sigma^*$ are* equivalent *to natural transformations $\lambda : \Sigma^*(\mathrm{Id} \times B) \Longrightarrow B\Sigma^*$ respecting the structure of Σ^* as follows:*

$$\begin{tikzcd}
\Sigma^*\Sigma^*(\mathrm{Id}\times B) \arrow[r, Rightarrow, "\Sigma^*\langle\Sigma^*\pi_1,\lambda\rangle"] \arrow[d, Rightarrow, "\mu_{(\mathrm{Id}\times B)}"] & \Sigma^*(\mathrm{Id}\times B)\Sigma^* \arrow[r, Rightarrow, "\lambda_{\Sigma^*}"] & B\Sigma^*\Sigma^* \arrow[d, Rightarrow, "B\mu"] \\
\Sigma^*(\mathrm{Id}\times B) \arrow[rr, Rightarrow, "\lambda"] & & B\Sigma^*
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathrm{Id}\times B \arrow[d, Rightarrow, "\eta_{(\mathrm{Id}\times B)}"'] \arrow[dr, Rightarrow, "B\eta\circ\pi_2"] & \\
\Sigma^*(\mathrm{Id}\times B) \arrow[r, Rightarrow, "\lambda"] & B\Sigma^*
\end{tikzcd}$$

▶ Remark 2.3. We shall be calling both GSOS laws ρ and natural transformations λ, as used in Proposition 2.2, *GSOS laws* for the sake of brevity.

GSOS laws induce *bialgebras*, i.e. algebra-coalgebra pairs that agree with the semantics.

▶ **Definition 2.4.** *A* bialgebra *for a GSOS law $\lambda : \Sigma^*(\mathrm{Id} \times B) \Longrightarrow B\Sigma^*$ (resp. $\rho : \Sigma(\mathrm{Id} \times B) \Longrightarrow B\Sigma^*$) is a Σ-algebra, B-coalgebra pair $\Sigma X \xrightarrow{g} X \xrightarrow{h} BX$ that commutes with λ (ρ):*

$$\begin{tikzcd}
\Sigma^*X \arrow[r, "g^\#"] \arrow[d, "\Sigma^*\langle 1,h\rangle"] & X \arrow[r, "h"] & BX \\
\Sigma^*(X\times BX) \arrow[rr, "\lambda"] & & B\Sigma^*X \arrow[u, "Bg^\#"']
\end{tikzcd}
\quad
\begin{tikzcd}
\Sigma X \arrow[r, "g"] \arrow[d, "\Sigma\langle 1,h\rangle"] & X \arrow[r, "h"] & BX \\
\Sigma(X\times BX) \arrow[rr, "\rho"] & & B\Sigma^*X \arrow[u, "Bg^\#"']
\end{tikzcd}$$

Where $g^\#$ is the $\mathcal{EM}$-algebra induced by g. A bialgebra morphism *from $\Sigma X \xrightarrow{g} X \xrightarrow{h} BX$ to $\Sigma Y \xrightarrow{j} Y \xrightarrow{k} BY$ is a map $f : X \to Y$ that is both a Σ-algebra and a B-coalgebra morphism.*

If $a : \ \Sigma A \xrightarrow{\cong} A$ is the initial Σ-algebra, the algebra of terms, and $z : \ Z \xrightarrow{\cong} BZ$ is the final B-coalgebra, the coalgebra of behaviors, the following proposition promotes a GSOS law to an *operational semantics* of a language, in the form of a morphism f mapping programs living in A to behaviors in Z.

▶ **Proposition 2.5** (From [17, 35]). *Let endofunctors $\Sigma, B : \mathbb{C} \to \mathbb{C}$ on a cartesian category $\mathbb{C}$ such that $a : \ \Sigma A \xrightarrow{\cong} A$ is the initial Σ-algebra and $z : \ Z \xrightarrow{\cong} BZ$ is the final B-coalgebra. Every GSOS law $\lambda : \Sigma^*(\mathrm{Id} \times B) \Longrightarrow B\Sigma^*$ induces a unique initial λ-bialgebra $\Sigma A \xrightarrow{a} A \xrightarrow{h} BA$, the* operational model *of the language, and a unique final λ-bialgebra $\Sigma Z \xrightarrow{g} Z \xrightarrow{z} BZ$, the* denotational model. *In addition, there exists a unique λ-bialgebra morphism $f : A \to Z$, mapping every program in A to its behavior in Z.*

The fact that map $f : A \to Z$ is an algebra homomorphism is a fundamental well-behavedness property of GSOS laws, as it implies that bisimilarity, defined as equality under f, is a *congruence*, i.e. is respected by all syntactic operators.

The categorical interpretation of GSOS rules can be better understood by the following examples, which serve as the "running" examples of this paper.

▶ **Example 2.6** (A *While* language). We introduce *While*, a basic imperative language with a mutable state, whose syntax is generated by the following grammar:

$\langle prog \rangle ::= \texttt{skip} \mid v \texttt{ := } \langle expr \rangle \mid \langle prog \rangle \texttt{ ; } \langle prog \rangle \mid \texttt{while } \langle expr \rangle \ \langle prog \rangle$

The statements are the standard `skip`, assignment `:=`, sequential composition `;` and `while`-loops. Expressions and assignments act on a variable store whose type we denote as S and we shall be writing $[e]_s$ to denote evaluation of an expression e under variable store s.

The syntax of *While* corresponds to the **Set**-endofunctor $\Sigma \triangleq \top \uplus (V \times Exp) \uplus \mathrm{Id}^2 \uplus (Exp \times \mathrm{Id})$, with the set of *While*-programs A as the carrier of the initial Σ-algebra $a : \Sigma A \xrightarrow{\cong} A$. The typical behavior functor for deterministic systems with mutable state is $[S \times (\{\checkmark\} \uplus \mathrm{Id})]^S$, where $\{\checkmark\} \cong \top$ is populated by the element that denotes *termination*, but we will instead use $T \triangleq [\mathcal{P}_c(S \times (\{\checkmark\} \uplus \mathrm{Id}))]^S$, where $\mathcal{P}_c$ is the *countable* power-set monad. This way the behavior can be equipped with both a monadic and an order structure, given by inclusion (see Section 2.2), while also allowing for non-determinism. The carrier of the final coalgebra $z : Z \xrightarrow{\cong} TZ$ is the set of behaviors acting on variable stores S returning countably many new stores and possibly new behaviors.

$$\text{skip}\,\frac{}{s, \texttt{skip} \to s, \checkmark} \qquad \text{asn}\,\frac{}{s, v \texttt{ := } e \to s_{[v \leftarrow [e]_s]}, \checkmark} \qquad \text{seq1}\,\frac{s, p \to s', \checkmark}{s, p \texttt{;} q \to s', q}$$

$$\text{seq2}\,\frac{s, p \to s', p'}{s, p \texttt{;} q \to s', p' \texttt{;} q} \qquad \text{w1}\,\frac{[e]_s = 0}{s, \texttt{while } e\ p \to s, \checkmark} \qquad \text{w2}\,\frac{[e]_s \neq 0 \quad q \triangleq \texttt{while } e\ p}{s, q \to s, p \ \texttt{;}\ q}$$

The above semantics determines a GSOS law $\rho : \Sigma(\mathrm{Id} \times T) \Longrightarrow T\Sigma^*$ in the category **Set** of sets and (total) functions, or equivalently a natural transformation $\lambda : \Sigma^*(\mathrm{Id} \times T) \Longrightarrow T\Sigma^*$. This example is covered in the literature [34, 33], but we include the definition for posterity.

▶ **Definition 2.7** (GSOS law of *While*).

$$\begin{array}{lll}
\rho_X : \Sigma(X \times TX) & \to & T\Sigma^* X \\
(x, f) \ ; \ (y, g) & \mapsto & \lambda s.(\{(s', y) \mid (s', \checkmark) \in f(s)\} \ \cup \ \{(s', (x' \ ; \ y)) \mid (s', x') \in f(s)\}) \\
\texttt{while } e \ (x, f) & \mapsto & \lambda \ s. \begin{cases} \{(s, (x \ ; \ \texttt{while } e \ x))\} & \text{if } [e]_s \neq 0 \\ \{(s, \checkmark)\} & \text{if } [e]_s = 0 \end{cases} \\
\texttt{skip} & \mapsto & \lambda s.\{(s, \checkmark)\} \\
v := e & \mapsto & \lambda s.\{(s_{[v \leftarrow ev]}, \checkmark)\} \text{ for } ev = [e]_s
\end{array}$$

To see how the semantics is connected to the GSOS law, consider rules seq1 and seq2, corresponding to the first line in ρ. Roughly, writing $s, p \to s', \checkmark$ denotes that $(s', \checkmark) \in f(s)$. The fact that $s, p \to s', \checkmark$ and $s, p \to s', q$ are rule premises is reflected in the construction of the set of transitions. Note also that q is used in the conclusion of rules seq1 and seq2. Similarly in ρ we can see that y is present in both the left and right side of ρ, which is why the shape of the behavior of subterms is $X \times TX$ rather than simply TX.

▶ **Example 2.8** (A simple process calculus). We introduce a simple process calculus, or *SPC* for short, based on the classic *Calculus of communicating systems* introduced by Milner [18]. Its syntax is generated by the following grammar:

$\langle p \rangle ::= \mathsf{0} \mid \delta.\langle p \rangle \mid \langle p \rangle \parallel \langle p \rangle \mid \langle p \rangle + \langle p \rangle$

From left to right we have the *null process* $\mathsf{0}$, *prefixing* of a process p with action $\delta \in \Delta_\tau = \Delta \cup \overline{\Delta} \cup \{\tau\}$ living in a set composed of actions Δ, coactions $\overline{\Delta}$ and an *internal* action τ, *parallel composition* and finally *non-deterministic choice*.

$$\text{sum1}\,\frac{P \xrightarrow{\delta} P'}{P + Q \xrightarrow{\delta} P'} \qquad \text{sum2}\,\frac{Q \xrightarrow{\delta} Q'}{P + Q \xrightarrow{\delta} Q'} \qquad \text{com1}\,\frac{P \xrightarrow{\delta} P'}{P \| Q \xrightarrow{\delta} P' \| Q}$$

$$\text{com2}\,\frac{Q \xrightarrow{\delta} Q'}{P \| Q \xrightarrow{\delta} P \| Q'} \qquad \text{syn}\,\frac{\alpha \in \Delta \quad P \xrightarrow{\alpha} P' \quad Q \xrightarrow{\overline{\alpha}} Q'}{P \| Q \xrightarrow{\tau} P' \| Q'} \qquad \text{prefix}\,\frac{}{\delta.P \xrightarrow{\delta} P}$$

Similarly to Example 2.6, the semantics forms a GSOS law $\rho : \Sigma(\mathrm{Id} \times T) \Longrightarrow T\Sigma^*$ in **Set** for $\Sigma \triangleq \top \uplus (\Delta_\tau \times \mathrm{Id}) \uplus \mathrm{Id}^2 \uplus \mathrm{Id}^2$ and $T \triangleq \mathcal{P}_c(\Delta_\tau \times \mathrm{Id})$. In this case, the carrier of the final coalgebra $z : \; Z \xrightarrow{\cong} TZ$ is the set of all strongly extensional (meaning that distinct children of nodes are not bisimilar), countably-branching, Δ_τ-labelled trees [2, §4].

▶ **Definition 2.9** (GSOS law of *SPC*).

$$\begin{array}{lll}
\rho_X : \Sigma(X \times TX) & \mapsto & T\Sigma^* X \\
\delta.(x, W) & \mapsto & \{(\delta, x)\} \\
(x, X) \parallel (y, Z) & \mapsto & \{(\delta, (x' \parallel y)) \mid (\delta, x') \in W\} \;\cup\; \{(\delta, (x \parallel y')) \mid (\delta, y') \in Z\} \cup \\
 & & \{(\tau, (x' \parallel y')) \mid (\alpha, x') \in W \wedge (\overline{a}, y') \in Z\} \\
(x, W) + (y, Z) & \mapsto & W \cup Z
\end{array}$$

2.2 Order-enrichment

Order-enriched categories [38] typically equip their hom-sets with an *order* structure. While there are many forms of order-enrichment, one "nice" such form that is convenient for our purposes is ω-$\mathbf{Cpo}^{\vee}_{ld}$-enrichment.

▶ **Definition 2.10** ([9, §2.3]). *A category $\mathbb{C}$ is ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched when*

- *Every hom-set $\mathbb{C}(X, Y)$ carries a partial order $\leq$ and has all finite joins $\vee$.*
- *Composition is left-distributive over finite joins, i.e. given any morphisms f, g, i with suitable domains and codomains, $i \circ (f \vee g) = i \circ f \vee i \circ g$.*
- *Every ascending ω-chain $f_0 \leq f_1 \leq \ldots$ for $f_i \in \mathbb{C}(X, Y)$ has a supremum $\bigvee_i f_i \in \mathbb{C}(X, Y)$.*
- *Composition $- \circ - : \mathbb{C}(X, Y) \times \mathbb{C}(Y, Z) \to \mathbb{C}(X, Z)$ is continuous, meaning that for any ascending ω-chains f_i, g_i and morphisms f, g with suitable (co)domains, we have*

$$g \circ \bigvee_i f_i = \bigvee_i (g \circ f_i) \quad \text{and} \quad (\bigvee_i g_i) \circ f = \bigvee_i (g_i \circ f)$$

For reasons that will become clear in Section 2.3, we are interested in monads whose Kleisli category is ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched as we are looking to use them as behaviors in our distributive laws and exploit their order structure. Examples of such monads are the powerset $\mathcal{P}$ [9, §4.1], the countable powerset $\mathcal{P}_c$ [9, §4.3] and the convex combination monad $\mathcal{CM}$ [9, §4.3].

▶ **Example 2.11** (Continuation of Example 2.6). The monad structure $\langle T, \eta, \mu \rangle$ for $T \triangleq [\mathcal{P}_c(S \times (\{\checkmark\} \uplus \mathrm{Id}))]^S$ is given by $\eta(x)(s) = \{(s, x)\}$ and $\mu(f)(s) = \bigcup_{(s',g) \in f(s)} \begin{cases} (s', \checkmark) & \text{if } g = \checkmark \\ g(s') & \text{if } g \neq \checkmark \end{cases}$.

The ω-$\mathbf{Cpo}^{\vee}_{ld}$-enrichment of $\mathcal{K}\ell(T)$ stems from the fact that the countable powerset monad $\mathcal{P}_c$ is itself ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched[1]. More specifically, we have $f \le g \iff \forall x, s. f(x)(s) \subseteq g(x)(s)$, $f \vee g \triangleq \lambda x.\lambda s.[f(x)(s) \cup g(x)(s)]$ and $\bigvee_i f_i = \lambda x.\lambda s. \bigcup_i f_i(x)(s)$.

▶ **Example 2.12** (Continuation of Example 2.8)**.** The monad structure $\langle T, \eta, \mu\rangle$ for $T \triangleq \mathcal{P}_c(\Delta_\tau \times \mathrm{Id})$ was developed by Brengos [8, §4.1] as a central, demonstrative example of the role of monads in coalgebraic weak bisimulation. The unit is simply $\eta(x) = \{(\tau, x)\}$ and the join $\mu_X : \mathcal{P}_c[\Delta_\tau \times \mathcal{P}_c(\Delta_\tau \times X)] \to \mathcal{P}_c(\Delta_\tau \times X)$ is $\mu = \mu_{\mathcal{P}_c} \circ \mathcal{P}_c\mu_\Delta \circ \mu_{\mathcal{P}_c} \circ \mathcal{P}_c st$ where $st_{X,Y} : X \times \mathcal{P}_c Y \to \mathcal{P}_c(X \times Y)$ is the *tensorial strength* of $\mathcal{P}_c$ given by

$$st_{X,Y}(x, Y) = \{(x, y) \mid y \in Y\}$$

and $\mu_\Delta : \Delta_\tau \times \Delta_\tau \times X \to \mathcal{P}_c\Delta_\tau X$ is

$$\mu_\Delta = \begin{cases} (\delta, \tau, x) \mapsto \{(\delta, x)\} \\ (\tau, \delta, x) \mapsto \{(\delta, x)\} \\ (\delta_1, \delta_2, x) \mapsto \emptyset \quad \text{when} \quad \delta_1, \delta_2 \neq \tau. \end{cases}$$

In other words, μ will disallow any two-step transition that outputs two visible labels in a row and allow everything else, but only after redundant invisible labels are removed. The motivation behind the definition of μ will become clear in Section 2.3. As for the ω-$\mathbf{Cpo}^{\vee}_{ld}$-enrichment of $\mathcal{K}\ell(T)$, it is also a consequence of $\mathcal{K}\ell(\mathcal{P}_c)$ being ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched, with $f \le g \iff f(x) \subseteq g(x)$, $f \vee g \triangleq \lambda x.[f(x) \cup g(x)]$ and $\bigvee_i f_i = \lambda x. \bigcup_i f_i(x)$.

2.3 Free monads in ω-$\mathrm{Cpo}^{\vee}_{ld}$-enriched categories

We now turn our attention to the coalgebraic approach to weak bisimulation of Brengos [8, 9]. The main idea is that given an endomorphism $\alpha : X \to X$ in an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched category, there exists the *free monad* over α, α^* . This process, which we call the *reflexive transitive closure* of a or simply the *rt-closure* of a, can be used to derive saturated transition systems or the *multi-step* evaluation relation of a programming language.

▶ Remark 2.13. In general, a *monad* in a bicategory K is an endomorphism $\epsilon : X \to X$ equipped with 2-cells $\eta : 1_X \to \epsilon$ and $\mu : \epsilon \circ \epsilon \to \epsilon$ subject to the conditions $\mu \circ \epsilon\eta = \mu \circ \eta\epsilon = 1_\epsilon$ and $\mu \circ \epsilon\mu = \mu \circ \mu\epsilon$. One can recover the classic definition of a monad by taking the strict 2-category $\mathbf{Cat}$ of categories, functors and natural transformation as K. In order-enriched categories, where there is at most one 2-cell between morphisms, usually denoted by $\le$, monads are simply endomorphisms $\epsilon : X \to X$ satisfying $1_X \le \epsilon$ and $\epsilon \circ \epsilon \le \epsilon$. Monads in order-enriched categories are known as *closure operators.*

▶ **Definition 2.14** ([9, Definition 3.6])**.** *An order-enriched category* $\mathbb{K}$ admits free monads *if for any endomorphism* $\alpha : X \to X$ *there exists a monad* α^* *such that*

- $\alpha \le \alpha^*$
- *if* $\alpha \le \beta$ *for a monad* $\beta : X \to X$ *then* $\alpha^* \le \beta$.

The free monad of an endomorphism in ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched categories is the least solution of a certain assignment or, equivalently, the supremum of an ascending ω-chain.

▶ **Proposition 2.15** ([9, §3.2])**.** *For an* ω-$\mathbf{Cpo}^{\vee}_{ld}$*-enriched category* $\mathbb{C}$*, the free monad* $(-)^* : \forall X.\mathbb{C}(X, X) \to \mathbb{C}(X, X)$ *of* $\alpha : X \to X$ *is given by* $\alpha^* \triangleq \mu x.1 \vee x \circ \alpha = \bigvee_{n<\omega}(1 \vee \alpha)^n$.

[1] We (slightly abusively) say that a monad T is ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched whenever $\mathcal{K}\ell(T)$ is.

Free monads in ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched categories enjoy a number of interesting properties.

▶ **Lemma 2.16** (Properties of free monads). *For all $\alpha : X \to X$, $\beta : Y \to Y$, we have*

i. $\forall f : X \to Y.\ f \circ \alpha \leq \beta \circ f \implies f \circ \alpha^* \leq \beta^* \circ f$
ii. $\alpha^{**} = \alpha^*$
iii. $\alpha^* = \alpha^* \circ \alpha^*$
iv. $1^* = 1$.

We can now revisit Example 2.6 and Example 2.8 to witness how the rt-closure acts on each operational model $h : A \to TA$. Note that throughout the paper we shall be using the notation $- \diamond -$ to denote composition in Kleisli categories.

▶ **Example 2.17** (Continuation of Example 2.6). The rt-closure (free monad) h^* of the operational model $h : A \to TA$ amounts to the reflexive, transitive closure of h. The initial stage $(1 \vee h)^0 = 1_{\mathcal{K}\ell(T)} = \eta_A$ takes care of the *reflexive* step, while stages $(1 \vee h)^{n+1} = (1 \vee h)^n \diamond (1 \vee h) = (1 \vee h)^n \vee ((1 \vee h)^n \diamond h)$ amount to the inductive, *transitive* step, which acts according to the definition of monadic composition. For instance, (s, `skip ; skip`) weakly transitions to (s, `skip ; skip`), (s, `skip`) and (s , ✓).

▶ **Example 2.18** (Continuation of Example 2.8). The rt-closure h^* amounts to the saturation of $h : A \to TA$. The unit η establishes $p \xRightarrow{\tau} p$ as a silent step in h^* for $p \in A$ whereas μ ensures that one-step transitions in the saturated system *cannot* produce more than one visible label, i.e. they will always be of the sort of $p \xRightarrow{\tau^*} q \xrightarrow{\delta} r \xRightarrow{\tau^*} s$ or $p \xRightarrow{\tau^*} q$. The reader may refer to [8, 9] for more details.

2.4 Weak bisimulation

We are now ready to define *weak bisimulation* as a special case of an *Aczel-Mendler bisimulation* [1, 32].

▶ **Definition 2.19.** *Let $f : X \to FX$ be a coalgebra for a functor $F : \mathbb{C} \to \mathbb{C}$. An* Aczel-Mendler bisimulation, *or simply* bisimulation, *for f is a relation (span) $X \xleftarrow{r_1} R \xrightarrow{r_2} X$ which is the carrier of a coalgebra $e : R \to FR$ that lifts to a span of coalgebra homomorphisms, i.e. making the following diagram commute.*

$$\begin{tikzcd}
FX & FR \arrow[l, "Fr_1"'] \arrow[r, "Fr_2"] & FX \\
X \arrow[u, "f"] & R \arrow[l, "r_1"] \arrow[r, "r_2"'] \arrow[u, "e"] & X \arrow[u, "f"]
\end{tikzcd}$$

In our setting we elect to use the "unoptimized" definition of weak bisimulation as bisimulation on a saturated system, mainly due to its simplicity. Regardless, under the mild condition that *arbitrary cotupling in $\mathcal{K}\ell(T)$ is monotonic*, which is true in all of our examples, this definition of weak bisimulation coincides with the traditional one [8, §6].

▶ **Definition 2.20.** *Assume a coalgebra $h : X \to TX$ for an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched monad $\langle T, \eta, \mu \rangle$. A* weak bisimulation *for h is a bisimulation for h^*.*

Two elements $x, y \in X$ are *(weakly) bisimilar*, written as $(x \approx y)$ $x \sim y$, if there is a (weak) bisimulation that contains them. If $z : Z \to TZ$ is the final T-coalgebra, then for every coalgebra $h : X \to TX$ we write $h^! : X \to Z$ for the unique coalgebra homomorphism from h to z and $h^\dagger$ for the one from h^* to z. The following theorem presents the principle of weak coinduction, i.e. that weakly bisimilar elements are mapped to the same weak behavior.

▶ **Theorem 2.21** ([8, Theorem 6.8]). *Let $\langle T, \eta, \mu \rangle$ be an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched monad with final coalgebra $z : Z \to TZ$. If T preserves weak pullbacks, then the greatest weak bisimulation for a coalgebra $h : X \to TX$ exists and coincides with the pullback of the equality span $\langle 1_Z, 1_Z \rangle : Z \rightarrowtail Z \times Z$ along $h^{\dagger} \times h^{\dagger} : X \times X \to Z \times Z$.*

The † construction can be applied to any coalgebra, including the final coalgebra $z : Z \to TZ$. The following lemma shows that $h^{\dagger}$ can also be recovered via $z^{\dagger}$ and $h^{!}$, a fact that will turn out to be useful in Section 3.

▶ **Lemma 2.22** ([8, Lemma 6.9]). *For* any $h : X \to TX$*, we have* $h^{\dagger} = z^{\dagger} \circ h^{!}$.

3 Weak bisimulation congruence semantics

The theory introduced in Section 2.2 sets the stage for our three compositionality criteria, which ensure that weak bisimilarity is a congruence. In addition, we shall test the criteria on *While* and *SPC* and establish a correspondence with the simply WB cool rule format of van Glabbeek (see [37], definition also included in Appendix A), both formally and through examples, as it also guarantees weak bisimilarity being a congruence and comes with less overhead compared to the WB cool format. Finally, we shall briefly touch on the work of Bonchi et al. [6] and show that systems satisfying the three criteria induce lax models.

3.1 The three compositionality criteria

Simply put, the three criteria ensure that semantics interact with the order structure of the behavior functor in a sensible way. They are *abstract* in that they apply to any GSOS law $\lambda : \Sigma^{*}(\mathrm{Id} \times T) \Longrightarrow T\Sigma^{*}$ when $\langle T, \eta, \mu \rangle$ is an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched monad. As such, we assume the existence of such a T and λ throughout Section 3. We present and explore each criterion individually starting with the simplest of the three.

▶ **Criterion 1** (Continuity). *For any ascending ω-chain $f_0 \leq f_1 \leq \ldots : X \to TX$ the following condition applies:*

$$\lambda \circ \Sigma^{*}\langle 1, \bigvee_i f_i \rangle = \bigvee_i \lambda \circ \Sigma^{*}\langle 1, f_i \rangle \tag{1}$$

Alternatively, we can write the above using $\rho : \Sigma(\mathrm{Id} \times T) \Longrightarrow T\Sigma^{*}$ as

$$\rho \circ \Sigma\langle 1, \bigvee_i f_i \rangle = \bigvee_i \rho \circ \Sigma\langle 1, f_i \rangle. \tag{2}$$

▶ **Proposition 3.1.** *Equation* (1) *and Equation* (2) *are equivalent.*

We can compare Criterion 1 to the *local continuity* property of lifted functors in Kleisli categories [14, §2.3], i.e. lifted functors respecting the ω-$\mathbf{Cpo}^{\vee}_{ld}$-enrichment structure of the category. When it comes to semantics, the following example from van Glabbeek [37, Example 2] underlines what sort of rules may violate Criterion 1.

▶ **Example 3.2** (Illegal rules). Let us extend *SPC* with a new unary operator, $\lfloor\langle p \rangle\rfloor$, subject to the following GSOS rules applying to specific actions $a, b \in \Delta$:

$$\text{pos}\,\frac{P \xrightarrow{\tau} P'}{\lfloor P \rfloor \xrightarrow{\tau} \lfloor P' \rfloor} \qquad \text{neg}\,\frac{P \stackrel{a}{\not\longrightarrow}}{\lfloor P \rfloor \xrightarrow{b} 0}$$

Rule neg allows a process that cannot perform an a-transition to terminate with a b-transition. This rule violates Criterion 1, which can be witnessed by testing the criterion with $(f_0(p) = \{(\tau, p')\}) \leq (f_1(p) = \{(a, q), (\tau, p')\})$, as only f_0 is able to induce a b-transition. In addition, weak bisimulation fails to be a congruence as $\tau.\alpha.\mathsf{0} \approx \alpha.\mathsf{0}$ but $\lfloor \tau.\alpha.\mathsf{0} \rfloor \not\approx \lfloor \alpha.\mathsf{0} \rfloor$.

Rule neg in the above example includes a negative premise. A GSOS specification that has no negative premises - conclusions are never negative - is called *positive.* Positive GSOS specifications correspond to *monotone* GSOS laws (see [11] and also [6, Equation 7]), in the sense that $g \leq f \implies \lambda \circ \Sigma^*\langle 1,g\rangle \leq \lambda \circ \Sigma^*\langle 1,f\rangle$. This is weaker than continuity (Criterion 1), in that continuous GSOS laws are monotone but not the other way around, but one has to look very hard to find semantics that are monotone yet not continuous.

▶ **Example 3.3** (A non-continuous monotone rule)**.** We substitute the set of visible actions of *SPC* with $\mathbb{N}$ and define a new operator, $\lfloor \_ \rfloor$, subject to rule mon $\dfrac{P \xrightarrow{\infty}}{\lfloor P \rfloor \xrightarrow{\tau} \mathsf{0}}$, where $P \xrightarrow{\infty}$ denotes that P can perform an infinite number of transitions, i.e. set $\{(\delta, P') \mid P \xrightarrow{\delta} P'\}$ is infinite. Even though this is a monotone rule, it is not continuous. Consider for instance the ascending ω-chain $f_i : X \to TX$ for some set of processes X with $f_i(x) = \{(0, x), (1, x) \ldots (i, x)\}$. Notice that $(\tau, x) \subseteq \lambda \circ \Sigma^*\langle 1, \bigvee_i f_i\rangle$, but $(\tau, x) \not\subseteq \bigvee_i \lambda \circ \Sigma^*\langle 1, f_i\rangle$.

▶ **Example 3.4** (Continuation of Example 2.6)**.** Back to our *While* language, Criterion 1 is trivially true for all terms except for sequential composition, which is slightly more involved. In this case, we can see that $\rho_X((x, \bigvee_i f_i(x))\ ;\ (y, \bigvee_i f_i(y)))$ is mapped to

$$\begin{aligned}
&\lambda s.(\{(s', y) \mid (s', \checkmark) \in (\bigvee_i f_i(x))(s)\} \ \cup\ \{(s', (x'\ ;\ y)) \mid (s', x') \in (\bigvee_i f_i(y))(s)\}) \\
&\qquad = \lambda s. \bigvee_i (\{(s', y) \mid (s', \checkmark) \in f_i(x)(s)\} \ \cup\ \{(s', (x'\ ;\ y)) \mid (s', x') \in f_i(y)(s)\}),
\end{aligned}$$

which is precisely $\bigvee_i(\rho_X((x, f_i(x))\ ;\ (y, f_i(y))))$.

▶ **Example 3.5** (Continuation of Example 2.8)**.** The transitions of prefix expressions such as δP are, for any given P, independent of (the transitions of) P thus the prefix rule is trivially continuous. Transitions for parallel composition and non-deterministic choice are basically unions of transitions of their subterms and hence satisfy Criterion 1.

▶ **Criterion 2** (Unitality)**.** *For any* $f : X \to TX$*,*

$$\begin{tikzcd}
\Sigma^* X \arrow[d, "\Sigma^*\langle 1,\eta_X \vee f\rangle"'] \arrow[dr, bend left, "(\lambda_X \circ \Sigma^*\langle 1,f\rangle)^*"] \arrow[dr, phantom, "\leq"] & \\
\Sigma^*(X \times TX) \arrow[r, "\lambda_X"'] & T\Sigma^* X
\end{tikzcd}$$

This criterion characterizes how the semantics deal with internal steps, here represented by the monadic unit η. The right path on the diagram represents the rt-closed (weak) transitions of a composite term, the subterms of which (strongly) transition according to f. On the other hand, the left path represents the strong transitions of a composite term, the subterms of which may also perform internal steps. This criterion, which somewhat resembles the identity condition for lifting functors to Kleisli categories [14, §2.2], dictates that adding arbitrary internal steps to subterms should not lead to extraneous, meaningful observations.

▶ Remark 3.6. A slightly stronger but simpler formulation of Criterion 2 based on ρ is

$$\begin{tikzcd}
\Sigma X \arrow[d, "\Sigma\langle 1,\eta_X \vee f\rangle"'] \arrow[dr, bend left, "(\rho_X \circ \Sigma\langle 1,f\rangle)\vee(\eta_X \circ \theta_X)"] \arrow[dr, phantom, "\leq"] & \\
\Sigma(X \times TX) \arrow[r, "\rho_X"'] & T\Sigma^* X
\end{tikzcd}$$

where $\theta : \Sigma \Longrightarrow \Sigma^*$ is the universal natural transformation sending Σ to its free monad. Compared to the original criterion, which asks for the transitions induced by internal steps to *eventually* appear on the right side, this version asks for said transition to appear either on *step 0*, the *reflexive*, identity step (hence the added $\eta_X \circ \theta_X$), or *step 1*, i.e. the transitions induced immediately by f. We can apply similar logic to Proposition 3.1 to show that this version of Criterion 2 is stronger.

Criterion 2 works in the same way as the "patience rule" requirement of the simply WB cool rule format [37, Definition 8, item 2], which dictates that the only rules with τ-premises are patience rules. For instance, the patience rule for a unary operator $o(p)$ is $\dfrac{p \xrightarrow{\tau} p'}{o(p) \xrightarrow{\tau} o(p')}$. It is clear that patience rules are achieving the same effect of forcing composite terms to *only* relay silent steps of subterms.

▶ **Example 3.7** ([37, Example 4]). We extend *SPC* with $\lfloor\_\rfloor$ and introduce the following impatient rules:

$$\text{pat}\,\frac{P \xrightarrow{\tau} P'}{\lfloor P\rfloor \xrightarrow{\tau} \lfloor P'\rfloor} \qquad \text{imp}\,\frac{P \xrightarrow{\tau} P'}{\lfloor P\rfloor \xrightarrow{c} \lfloor P'\rfloor}$$

Rule imp violates Criterion 2, as taking $f = \lambda x.\varnothing$ will not induce the c-transition present in the left path. Weak bisimulation fails to be a congruence as $0 \approx \tau.0$ but $\lfloor 0\rfloor \not\approx \lfloor \tau.0\rfloor$.

▶ **Example 3.8** (Continuation of Example 2.6). Language *While* actually respects the stronger version of Criterion 2 found in Remark 3.6. For `skip`, assignment and `while`-loops the transitions induced by $\rho_X \circ \Sigma\langle 1,f\rangle$ are the same regardless of f. This is not the case for sequential composition, but we observe that the left path always leads to $s, x\,;y \to s, x\,;y$, which is covered by the added $\eta_X \circ \theta_X$ on the right path.

▶ **Example 3.9** (Continuation of Example 2.8). *SPC* provides for a good example of failure of Criterion 2 as it echoes the well-known fact that weak bisimilarity is not compatible with non-deterministic choice in a manner similar to Example 3.7. In particular, the left path always assigns $x+y$ transitions $\{(\tau, x), (\tau, y)\}$ but for $f = \lambda x.\varnothing$ we have $(\lambda_X \circ \Sigma^*\langle 1, f\rangle)^* = \{(\tau, x+y)\}$. Clearly $\{(\tau, x), (\tau, y)\} \nsubseteq \{(\tau, x + y)\}$ and so Criterion 2 is not satisfied. We can witness the incompatibility of non-deterministic choice by taking a cue from the failing instance and use a process which has no transitions, i.e. the null process: $0 \approx \tau.0$ but $\delta.0 + 0 \not\approx \delta.0 + \tau.0$.

▶ **Criterion 3** (Observability). *For any $f : X \to TX$,*

$$\lambda_X \circ \Sigma^*\langle 1,f \diamond f\rangle \leq (\lambda_X \circ \Sigma^*\langle 1,f\rangle)^* \tag{3}$$

Equivalently, we can reformulate the above as

$$\rho_X \circ \Sigma\langle 1,f \diamond f\rangle \leq (\lambda_X \circ \Sigma^*\langle 1,f\rangle)^* \circ \theta_X \tag{4}$$

where $\theta : \Sigma \Longrightarrow \Sigma^*$ is the universal natural transformation sending Σ to its free monad.

▶ Remark 3.10. The fact that Equation (3) and Equation (4) are equivalent can be proved in a manner similar to Proposition 3.1.

This criterion is roughly a weakening of the associativity condition for liftings to Kleisli categories [14, §2.2]. We can think of $f \diamond f$ as a two-step transition applied to subterms and $\lambda_X \circ \Sigma^*\langle 1, f \diamond f\rangle$ as the act of a context inspecting zero or more subterms performing that two-step transition. The criterion relates the information obtained by contexts when inspecting two-step transitions as opposed to inspecting each step individually, possibly many times over. Specifically, it ensures that the former always carries *less* information than the latter. A further way to interpret this criterion is that inspecting an effect *now* instead of *later* does not produce new outcomes.

Criterion 3 is more complex to explain in terms of the simply WB cool format, as requirements 1,3,4,5 in Definition 8 from van Glabbeek [37] all contribute towards observations on visible transitions not being affected by silent transitions, regardless of when the latter occur. First, let us look at *straightness*. An operator is straight if it has no rules where a variable occurs multiple times in the left-hand side of its premises. The following example shows how non-straight rules can lead to issues.

▶ **Example 3.11** ([37, Example 3]). Let operator $\lfloor\_\rfloor$ subject to the following rules applying to specific $a, b, c \in \Delta$:

$$\text{pat}\,\frac{P \xrightarrow{\tau} P'}{\lfloor P\rfloor \xrightarrow{\tau} \lfloor P'\rfloor} \qquad \text{cur}\,\frac{P \xrightarrow{a} Q \quad P \xrightarrow{b} W}{\lfloor P\rfloor \xrightarrow{c} \lfloor Q\rfloor}$$

We can see how rule cur violates Criterion 3 by taking $f(p) = \{(a, q), (\tau, w)\}$, $f(q) = \{(\tau, q)\}$ and $f(w) = \{(b, w)\}$. Since $(f \diamond f)(p) = \{(a, q), (b, w)\}$, running $\lambda_X \circ \Sigma^*\langle 1, f \diamond f\rangle$ on $\lfloor p\rfloor$ induces a c-transition, which does not occur on the right side. With rule cur weak bisimilarity is not a congruence, as $a.0 + b.0 + \tau.b.0 \approx a.0 + \tau.b.0$ but $\lfloor a.0 + b.0 + \tau.b.0\rfloor \not\approx \lfloor a.0 + \tau.b.0\rfloor$.

Requirements 3 and 4 in the definition of the simply WB cool format underline how the *lack* of patience rules can affect observations.

▶ **Example 3.12** ([37, Example 5]). Consider operator $\lfloor\_\rfloor$ subject to rule $\text{oba}\,\frac{P \xrightarrow{a} P'}{\lfloor P\rfloor \xrightarrow{\tau} 0}$ applying to a specific action a. Rule oba fails Criterion 3 with $f(p) = \{(\tau, q)\}$ and $f(q) = \{(a, w)\}$, making $(f \diamond f)(p) = \{(a, w)\}$. Running $\lambda_X \circ \Sigma^*\langle 1, f \diamond f\rangle$ on $\lfloor p\rfloor$ induces a τ-transition, which does not occur on the right side. We can see how $a.0 \approx \tau.a.0$ but $\lfloor a.0\rfloor \not\approx \lfloor \tau.a.0\rfloor$.

The final requirement is that of *smoothness*. A straight operator for an LTS is smooth if it has no rules where a variable occurs both in the target and in the left-hand side of a premise. Non-smooth rules can also cause problems, as evidenced by the following example.

▶ **Example 3.13** ([37, Example 7]). Let operator $\lfloor\_\rfloor$ with the following rules:

$$\text{play}\,\frac{P \xrightarrow{\delta} P'}{\lfloor P\rfloor \xrightarrow{\delta} \lfloor P'\rfloor} \qquad \text{pause}\,\frac{P \xrightarrow{\delta} P'}{\lfloor P\rfloor \xrightarrow{\delta} \lfloor P\rfloor}$$

Non-smooth rule pause also violates Criterion 3. Take $f(p) = \{(\tau, q)\}$ and $f(q) = \{(a, w)\}$, making $(f \diamond f)(p) = \{(a, w)\}$. Running $\lambda_X \circ \Sigma^*\langle 1, f \diamond f\rangle$ on $\lfloor p\rfloor$ induces a-transition $(a, \lfloor p\rfloor)$, but the only a-transitions induced on the other side are $(a, \lfloor q\rfloor)$ and $(a, \lfloor w\rfloor)$ instead. Weak bisimilarity fails to be a congruence, with $a.0 + \tau.b.0 \approx a.0 + \tau.b.0 + b.0$ but $\lfloor a.0 + \tau.b.0\rfloor \not\approx \lfloor a.0 + \tau.b.0 + b.0\rfloor$. The difference here is that only $\lfloor a.0 + \tau.b.0 + b.0\rfloor$ is able to perform an a-transition after a b-transition.

▶ **Example 3.14** (Continuation of Example 2.6)**.** The situation for Criterion 3 is less obvious for both `while`-loops and sequential composition, but still trivial for `skip` and assignment statements. Using the simpler ρ-based formulation of Criterion 3, we have that for `while`-loops, the induced transition is not affected by transitions of subterms, hence $\rho_X \circ \Sigma\langle 1, f \diamond f\rangle$ is always included in the first iteration of $(\lambda_X \circ \Sigma^*\langle 1, f\rangle)^* \circ \theta$.

Showing that the criterion is satisfied by an expression $x;y$ for any $x, y \in X$ requires case analysis of $(f \diamond f)(x)$ only, as the rule ignores y. The two cases are:

- $(t, z) \in (f \diamond f)(x)(s)$. In this case x did not terminate, but rather went through an intermediate transition t', z'. According to rule seq2, the transition produced by $\rho_X \circ \Sigma\langle 1, f \diamond f\rangle$ is $s, x;y \to t, z;y$. Going over to $(\lambda_X \circ \Sigma^*\langle 1, f\rangle)^* \circ \theta$, the first iteration produces $s, x;y \to t', z';y$, and in the second we get $t', z';y \to t, z;y$, which is the same result.
- $(t, \checkmark) \in (f \diamond f)(x)(s)$. Here, x terminated producing t either immediately ($(t, \checkmark) \in f(x)(s)$) or in two steps ($(s', x') \in f(x)(s)$ and $(t, \checkmark) \in f(x')(s')$). In any case, the transition produced by $\rho_X \circ \Sigma\langle 1, f \diamond f\rangle$ is $s, x;y \to t, y$. Depending on when x terminated, this transition will be "caught" in either the first or the second iteration of $(\lambda_X \circ \Sigma^*\langle 1, f\rangle)^* \circ \theta$.

▶ **Example 3.15** (Continuation of Example 2.8)**.** When instantiated to *SPC*, the criterion essentially asks if the act of "forgetting" invisible steps of subterms, as imposed by the rules of monadic composition in $\mathcal{K}\ell(T)$, gives rise to new transitions for a composite term. In most cases, this is evidently true; consider for instance the parallel composition of two terms $P\|Q$, for which $P \xrightarrow{\tau} P' \xrightarrow{\delta} P''$, i.e. $(\tau, P') \in f(P)$ and $(\delta, P'') \in f(P')$. Forgetting the invisible step gives $P \xrightarrow{\delta} P''$ ($(\delta, P'') \in (f \diamond f)(P)$), so by rule com1 we have $P\|Q \xrightarrow{\delta} P''\|Q$ on the left-hand side. This transition will occur after two iterations on the right-hand side, as in $P\|Q \xrightarrow{\tau} P'\|Q \xrightarrow{\delta} P''\|Q$.

Rule syn is especially interesting, as it showcases the full power of Criterion 3. There are four different cases where syn induces a transition $P\|Q \xrightarrow{\tau} P''\|Q''$ on the left side, namely

- $P \xrightarrow{\tau} P' \xrightarrow{\alpha} P''$ and $Q \xrightarrow{\tau} Q' \xrightarrow{\overline{\alpha}} Q''$
- $P \xrightarrow{\tau} P' \xrightarrow{\alpha} P''$ and $Q \xrightarrow{\overline{\alpha}} Q' \xrightarrow{\tau} Q''$
- $P \xrightarrow{\alpha} P' \xrightarrow{\tau} P''$ and $Q \xrightarrow{\overline{\alpha}} Q' \xrightarrow{\tau} Q''$
- $P \xrightarrow{\alpha} P' \xrightarrow{\tau} P''$ and $Q \xrightarrow{\tau} Q' \xrightarrow{\overline{\alpha}} Q''$.

The third and fourth case work similarly to the first and second (resp.) and so we focus on the latter. Either way, the right side of Criterion 3 needs three iterations (meaning $(\lambda_X \circ \Sigma^*\langle 1, f\rangle) \diamond (\lambda_X \circ \Sigma^*\langle 1, f\rangle) \diamond (\rho_X \circ \Sigma\langle 1, f\rangle))$ in order to produce transition $P\|Q \xrightarrow{\tau} P''\|Q''$. In the first case, each iteration executes (in sequence) rules com1, com2 and syn leading to $P\|Q \xrightarrow{\tau} P'\|Q \xrightarrow{\tau} P'\|Q' \xrightarrow{\tau} P''\|Q''$. In the second case the order changes with rules com1, syn and com2 inducing $P\|Q \xrightarrow{\tau} P'\|Q \xrightarrow{\tau} P''\|Q' \xrightarrow{\tau} P''\|Q''$.

Up to this point, the connection of our criteria with the simply WB cool format has remained informal. The following theorem turns this connection to a formal correspondence.

▶ **Theorem 3.16.** *Any language in the simply WB cool format satisfies the three compositionality criteria.*

The converse is not true, as for example any simple non-smooth rule satisfying the three criteria, such as $[x] \xrightarrow{c} x$, is not simply WB cool. A proof of Theorem 3.16 is provided in Appendix B.2.

3.2 An algebra for a $\dagger$

We are now ready to move on to the main theorem of our work, namely the existence of a compatible algebra structure for morphism $h^\dagger : A \to Z$ mapping every term in A to its weak behavior in Z. First, we present the following intermediate result that plays a catalytic role in the main theorem and is noteworthy in its own right.

▶ **Proposition 3.17.** *Let $\lambda : \Sigma^*(\mathrm{Id} \times T) \Longrightarrow T\Sigma^*$ be a GSOS law of Σ over T with T being an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched monad. If λ satisfies the* three compositionality criteria, *then for any T-coalgebra $f : X \to TX$ we have $(\lambda \circ \Sigma^*\langle 1, f^*\rangle)^* = (\lambda \circ \Sigma^*\langle 1, f\rangle)^*$.*

Proof. By antisymmetry on the order structure $\leq$ of T. To avoid unnecessary clutter, for the rest of the proof we shall be writing $\lambda \circ \Sigma^*\langle 1, f\rangle$ as $\overline{\Sigma} f$ and the 1 in $\overline{\Sigma}(1 \vee f)$ will stand for η, the identity morphism in $\mathcal{K}\ell(T)$.

- Via Criterion 1 and Lemma 2.16 we have $f \leq f^* \implies \overline{\Sigma} f \leq \overline{\Sigma} f^* \implies (\overline{\Sigma} f)^* \leq (\overline{\Sigma} f^*)^*$. In other words, $(\lambda \circ \Sigma^*\langle 1, f\rangle)^* \leq (\lambda \circ \Sigma^*\langle 1, f^*\rangle)^*$.
- We first show that $\overline{\Sigma}(1 \vee f)^n \leq (\overline{\Sigma} f)^*$ for all $n \in \mathbb{N}$ by induction on n. For $n = 0$ and $n = 1$ and by Criterion 1 (as $1 \leq (1 \vee f)$) and Criterion 2,

$$\overline{\Sigma}(1 \vee f)^0 = \overline{\Sigma}1 \leq \overline{\Sigma}(1 \vee f) = \overline{\Sigma}(1 \vee f)^1 \leq (\overline{\Sigma} f)^* \tag{5}$$

We now have to show that $\overline{\Sigma}((1 \vee f)^{n+1} \diamond (1 \vee f)) \leq (\overline{\Sigma} f)^*$ for some n by making use of the inductive hypothesis $\overline{\Sigma}(1 \vee f)^{n+1} \leq (\overline{\Sigma} f)^*$. To that end, we first note that $1 \leq (1 \vee f)$ and thus, due to ω-$\mathbf{Cpo}^{\vee}_{ld}$-enrichment, we have that for all $n \in \mathbb{N}$,

$$1 \vee f \leq (1 \vee f) \diamond (1 \vee f) \leq \cdots \leq (1 \vee f)^{n+1} \implies (1 \vee f)^{n+1} \diamond (1 \vee f) \leq (1 \vee f)^{n+1} \diamond (1 \vee f)^{n+1} \tag{6}$$

Next, by (6), Criterion 1 and Criterion 3,

$$\overline{\Sigma}((1 \vee f)^{n+1} \diamond (1 \vee f)) \leq \overline{\Sigma}((1 \vee f)^{n+1} \diamond (1 \vee f)^{n+1}) \leq (\overline{\Sigma}(1 \vee f)^{n+1})^* \tag{7}$$

The induction hypothesis gives $\overline{\Sigma}(1 \vee f)^{n+1} \leq (\overline{\Sigma} f)^*$ and so (7) becomes

$$\overline{\Sigma}((1 \vee f)^{n+1} \diamond (1 \vee f)) \leq (\overline{\Sigma}(1 \vee f)^{n+1})^* \leq (\overline{\Sigma} f)^{**} = (\overline{\Sigma} f)^* \tag{8}$$

Inequalities (8) and (5) complete the inductive proof that $\forall n \in \mathbb{N}.\ \overline{\Sigma}(1 \vee f)^n \leq (\overline{\Sigma} f)^*$. Finally, by Proposition 2.15 and Criterion 1:

$$\overline{\Sigma} f^* = \overline{\Sigma} \bigvee_{n<\omega} (1 \vee f)^n = \bigvee_{n<\omega} \overline{\Sigma}(1 \vee f)^n$$

We just proved that every link in the ω-chain is smaller than $(\overline{\Sigma} f)^*$, and thus $\overline{\Sigma} f^* \leq (\overline{\Sigma} f)^*$. By Definition 2.14, this becomes $(\overline{\Sigma} f^*)^* \leq (\overline{\Sigma} f)^*$, i.e. $(\lambda \circ \Sigma^*\langle 1, f^*\rangle)^* \leq (\lambda \circ \Sigma^*\langle 1, f\rangle)^*$.

Having shown both directions, we end up with $(\lambda \circ \Sigma^*\langle 1, f^*\rangle)^* = (\lambda \circ \Sigma^*\langle 1, f\rangle)^*$. ◀

In other words, the transition system of (arbitrarily deep) contexts with subterms in X is *weakly* equivalent that of contexts having access to all the weak transitions of subterms in X. Proposition 3.17 is what enables the formation of a compatible algebra structure for $h^\dagger : A \to Z$ by applying it to the final coalgebra $z : Z \xrightarrow{\cong} BZ$. It is currently unclear if the converse of Proposition 3.17 holds.

▶ **Theorem 3.18** (Main theorem). *Let $\lambda : \Sigma^*(\mathrm{Id} \times T) \Longrightarrow T\Sigma^*$ be a GSOS law of Σ over T with T being an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched monad. If λ satisfies the* three compositionality criteria, *then for $\Sigma A \xrightarrow{a} A \xrightarrow{h} TA$ and $\Sigma Z \xrightarrow{g} Z \xrightarrow{z} TZ$ as in Proposition 2.5, $h^\dagger$ is a Σ-algebra homomorphism from $a : \ \Sigma A \xrightarrow{\cong} A$ to $z^\dagger \circ g : \Sigma Z \to Z$.*

Proof. We first observe that the following diagram commutes due to finality of z (top left and bottom rectangle) and naturality of λ. Note that the horizontal lines are all T-coalgebras and $g^\#$ is the $\mathcal{EM}$-algebra induced by the denotational model g.

$$\begin{tikzcd}
\Sigma^* Z \arrow[r, "\Sigma^*\langle 1,z^*\rangle"] \arrow[d, "\Sigma^*(z^\dagger)"'] & \Sigma^*(Z \times TZ) \arrow[r, "\lambda"] \arrow[d, "\Sigma^*(\mathrm{Id}\times T)(z^\dagger)"] & T\Sigma^* Z \arrow[d, "T\Sigma^*(z^\dagger)"] \\
\Sigma^* Z \arrow[r, "\Sigma^*\langle 1,z\rangle"] \arrow[d, "g^\#"'] & \Sigma^*(Z \times TZ) \arrow[r, "\lambda"] & T\Sigma^* Z \arrow[d, "Tg^\#"] \\
Z \arrow[rr, "z", "\cong"'] & & TZ
\end{tikzcd}$$

Since rt-closing preserves T-coalgebra homomorphisms, rt-closing the T-coalgebras and finality of z gives us

$$\begin{tikzcd}
\Sigma^* Z \arrow[r, "(\lambda\circ\Sigma^*\langle 1,z^*\rangle)^*"] \arrow[d, "\Sigma^*(z^\dagger)"'] & T\Sigma^* Z \arrow[d, "T\Sigma^*(z^\dagger)"] \\
\Sigma^* Z \arrow[r, "(\lambda\circ\Sigma^*\langle 1,z\rangle)^*"] \arrow[d, "g^\#"'] & T\Sigma^* Z \arrow[d, "Tg^\#"] \\
Z \arrow[r, "z^*"] \arrow[d, "z^\dagger"'] & TZ \arrow[d, "Tz^\dagger"] \\
Z \arrow[r, "z", "\cong"'] & TZ
\end{tikzcd}$$

Via Proposition 3.17 we have $(\lambda \circ \Sigma^*\langle 1,z^*\rangle)^* = (\lambda \circ \Sigma^*\langle 1,z\rangle)^*$. Since all homomorphisms on final coalgebras are unique, we see that $z^\dagger \circ g^\# = z^\dagger \circ g^\# \circ \Sigma^*(z^\dagger)$, which in turn makes the following diagram commute:

$$\begin{tikzcd}
\Sigma Z \arrow[r, "\theta_Z"] \arrow[d, "\Sigma z^\dagger"'] & \Sigma^* Z \arrow[r, "g^\#"] \arrow[d, "\Sigma^* z^\dagger"'] & Z \arrow[d, "z^\dagger"] \\
\Sigma Z \arrow[r, "\theta_Z"] & \Sigma^* Z \arrow[r, "z^\dagger\circ g^\#"] & Z
\end{tikzcd}$$

Where $\theta : \Sigma \implies \Sigma^*$ is the universal natural transformation sending Σ to its free monad. But $g^\# \circ \theta = g$ and so we can conclude

$$
\begin{aligned}
& z^\dagger \circ g && = && z^\dagger \circ g \circ \Sigma z^\dagger && (9)\\
\implies & z^\dagger \circ g \circ \Sigma h^! && = && z^\dagger \circ g \circ \Sigma z^\dagger \circ \Sigma h^! && (10)\\
\implies & z^\dagger \circ h^! \circ a && = && z^\dagger \circ g \circ \Sigma(z^\dagger \circ h^!) && (11)\\
\implies & h^\dagger \circ a && = && z^\dagger \circ g \circ \Sigma h^\dagger. && (12)
\end{aligned}
$$

Equation (10) gives (11) due to $h^!$ being a bialgebra morphism and finally we have (12) by Lemma 2.22, which finishes the proof. ◀

Weak bisimilarity being a congruence is thus a simple corollary of Theorems 2.21 and 3.18.

▶ **Corollary 3.19.** *Let $\lambda : \Sigma^*(\mathrm{Id} \times T) \implies T\Sigma^*$ be a GSOS law of Σ over weak-pullback preserving functor T with T being an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched monad as in Theorem 3.18. If λ satisfies the* three compositionality criteria*, weak bisimilarity in λ is a congruence.*

3.3 Lax models for weak bisimulation

Up-to techniques for bisimulation [19, 28] are techniques that simplify reasoning about behavioral equivalences, the main idea being that instead of having to show that two processes are included in a bisimulation relation, one can show that they are included in a different relation which is *sound* with respect to bisimulation. Bonchi et al. [6, Corollary 22] proved that weak bisimulation up-to contextual closure is a compatible (and hence sound) up-to technique for systems specified in the simply WB cool rule format. This result is a corollary of their main theorem [6, Theorem 20], which requires the underlying system to be *positive* and also the saturated transition system to be a *lax model* for the given specification, requirements that are automatically true for systems in the simply WB cool format.

As mentioned in Section 3.1, the abstract equivalent of positivity for GSOS specifications is monotonicity, which is weaker than continuity (Criterion 1). Thus, a GSOS law satisfying the three compositionality criteria is monotone. As for lax models, our behaviors come with an order structure and hence we can give the following definition.

▶ **Definition 3.20** (Lax models for GSOS laws). *Let $\lambda : \Sigma^*(\mathrm{Id} \times T) \Longrightarrow T\Sigma^*$ be a GSOS law of Σ over T with T being an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched monad. A* lax λ-model *is a Σ-algebra and T-coalgebra pair $\Sigma X \xrightarrow{g} X \xrightarrow{h} TX$ making the following diagram commute laxly:*

$$\begin{tikzcd}
\Sigma^* X \arrow[r, "g^{\#}"] \arrow[d, "\Sigma^*\langle 1,h\rangle"'] & X \arrow[r, "h"] \arrow[d, phantom, "\mathrel{\rotatebox{90}{$\leq$}}"] & TX \\
\Sigma^*(X \times TX) \arrow[rr, "\lambda"] & & T\Sigma^* X \arrow[u, "Tg^{\#}"']
\end{tikzcd}$$

Where $g^{\#}$ is the respective $\mathcal{EM}$-algebra induced by g.

In other words, a lax model is a relaxed version of a bialgebra (Definition 2.4), implying that *only*, but not necessarily *all*, weak transitions of a composite term can be deduced from the weak transitions of its subterms. One non-example of a lax model is $\Sigma A \xrightarrow{a} A \xrightarrow{h^*} BA$ of *SPC* from Example 2.8. We can use the same problematic case as Example 3.9: the lower path on the bialgebra diagram for process $\delta.0 + 0$ reveals transition $\delta.0 + 0 \xrightarrow{\tau} 0$, which is not a transition of $\delta.0 + 0$.

For all their nice properties, there is little indication as to which GSOS laws have lax models and why. Thus, in the context of our work, it is sensible to ask if the three congruence criteria are adequate with respect to producing lax models for GSOS laws, with the following theorem asserting that this is indeed the case.

▶ **Theorem 3.21.** *Let $\lambda : \Sigma^*(\mathrm{Id} \times T) \Longrightarrow T\Sigma^*$ be a GSOS law of Σ over T with T being an ω-$\mathbf{Cpo}^{\vee}_{ld}$-enriched monad. If λ satisfies the* three compositionality criteria*, then for* any *λ-bialgebra $\Sigma X \xrightarrow{g} X \xrightarrow{h} TX$, $\Sigma X \xrightarrow{g} X \xrightarrow{h^*} TX$ is a lax λ-model.*

Proof. By Lemma 2.16 and the definition of a bialgebra we have $Tg^{\#} \circ (\lambda_X \circ \Sigma^*\langle 1,h\rangle)^* = h^* \circ g^{\#}$. Thus, to prove that $\Sigma X \xrightarrow{g} X \xrightarrow{h^*} TX$ is lax λ-model, it suffices to show that $\lambda \circ \Sigma^*\langle 1,h^*\rangle \leq (\lambda \circ \Sigma^*\langle 1,h\rangle)^*$. by Proposition 3.17 and Definition 2.14 we indeed have that $\lambda \circ \Sigma^*\langle 1,h^*\rangle \leq (\lambda \circ \Sigma^*\langle 1,h^*\rangle)^* = (\lambda \circ \Sigma^*\langle 1,h\rangle)^*$. ◀

It is worth noting that compatibility of the up-to context technique for weak bisimulation entails the congruence property of weak bisimilarity, which means that there is an alternative route to Corollary 3.19 via Proposition 3.17 and [6, Theorem 20]. With that in mind, can lax models work as a formal method for proving congruence of weak bisimilarity like our three criteria? The problem is that proving laxness of a model (typically the initial, rt-closed model $\Sigma A \xrightarrow{a} A \xrightarrow{h^*} TA$) involves non-trivial reasoning on an rt-closed system that is itself defined inductively on the structure of terms. Conversely, our three criteria are significantly easier to establish as they characterize a GSOS law acting on a single layer of syntax.

4 Conclusion

In this paper we presented three abstract criteria over operational semantics, given in the form of Turi and Plotkin's bialgebraic semantics, that guarantee weak bisimilarity being a congruence. We believe that the criteria gracefully balance between generality and usefulness but, as is often the case with abstract results, this is something hard to assess accurately. What is equally important however is that each of the criteria can be given an intuitive explanation as to the kind of restriction it imposes on the semantics. We hope that these insights can contribute towards a conclusive answer to the general problem of full abstraction: the definition of the best adequate denotational semantics, the underlying equivalence of which coincides with contextual equivalence.

References

1 Peter Aczel and Nax Paul Mendler. A final coalgebra theorem. In David H. Pitt, David E. Rydeheard, Peter Dybjer, Andrew M. Pitts, and Axel Poigné, editors, *Category Theory and Computer Science, Manchester, UK, September 5–8, 1989, Proceedings*, volume 389 of *Lecture Notes in Computer Science*, pages 357–365. Springer, 1989. doi:10.1007/BFb0018361.

2 Jirí Adámek, Paul Blain Levy, Stefan Milius, Lawrence S. Moss, and Lurdes Sousa. On final coalgebras of power-set functors and saturated trees – to george janelidze on the occasion of his sixtieth birthday. *Applied Categorical Structures*, 23(4):609–641, 2015. doi:10.1007/s10485-014-9372-9.

3 Christel Baier and Holger Hermanns. Weak bisimulation for fully probabilistic processes. In Orna Grumberg, editor, *Computer Aided Verification, 9th International Conference, CAV '97, Haifa, Israel, June 22-25, 1997, Proceedings*, volume 1254 of *Lecture Notes in Computer Science*, pages 119–130. Springer, 1997. doi:10.1007/3-540-63166-6_14.

4 Bard Bloom. Structural operational semantics for weak bisimulations. *Theor. Comput. Sci.*, 146(1&2):25–68, 1995. doi:10.1016/0304-3975(94)00152-9.

5 Filippo Bonchi, Daniela Petrisan, Damien Pous, and Jurriaan Rot. Coinduction up-to in a fibrational setting. In Thomas A. Henzinger and Dale Miller, editors, *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14–18, 2014*, pages 20:1–20:9. ACM, 2014. doi:10.1145/2603088.2603149.

6 Filippo Bonchi, Daniela Petrisan, Damien Pous, and Jurriaan Rot. Lax bialgebras and up-to techniques for weak bisimulations. In Luca Aceto and David de Frutos-Escrig, editors, *26th International Conference on Concurrency Theory, CONCUR 2015, Madrid, Spain, September 1.4, 2015*, volume 42 of *LIPIcs*, pages 240–253. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.CONCUR.2015.240.

7 Filippo Bonchi, Daniela Petrisan, Damien Pous, and Jurriaan Rot. A general account of coinduction up-to. *Acta Inf.*, 54(2):127–190, 2017. doi:10.1007/s00236-016-0271-4.

8 Tomasz Brengos. Weak bisimulation for coalgebras over order enriched monads. *Logical Methods in Computer Science*, 11(2), 2015. doi:10.2168/LMCS-11(2:14)2015.

9 Tomasz Brengos, Marino Miculan, and Marco Peressotti. Behavioural equivalences for coalgebras with unobservable moves. *J. Log. Algebraic Methods Program.*, 84(6):826–852, 2015. doi:10.1016/j.jlamp.2015.09.002.

10 Derek Dreyer, Amal Ahmed, and Lars Birkedal. Logical step-indexed logical relations. *Logical Methods in Computer Science*, 7(2), 2011. `doi:10.2168/LMCS-7(2:16)2011`.

11 Marcelo Fiore and Sam Staton. Positive structural operational semantics and monotone distributive laws. In *CMCS'10 Short Contributions*, 2010.

12 Sergey Goncharov and Dirk Pattinson. Coalgebraic weak bisimulation from recursive equations over monads. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming – 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, volume 8573 of *Lecture Notes in Computer Science*, pages 196–207. Springer, 2014. `doi:10.1007/978-3-662-43951-7_17`.

13 Andrew D. Gordon. Bisimilarity as a theory of functional programming. *Theor. Comput. Sci.*, 228(1-2):5–47, 1999. `doi:10.1016/S0304-3975(98)00353-3`.

14 Ichiro Hasuo, Bart Jacobs, and Ana Sokolova. Generic trace semantics via coinduction. *Logical Methods in Computer Science*, 3(4), 2007. `doi:10.2168/LMCS-3(4:11)2007`.

15 Douglas J. Howe. Equality in lazy computation systems. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science (LICS '89), Pacific Grove, California, USA, June 5-8, 1989*, pages 198–203. IEEE Computer Society, 1989. `doi:10.1109/LICS.1989.39174`.

16 Douglas J. Howe. Proving congruence of bisimulation in functional programming languages. *Inf. Comput.*, 124(2):103–112, 1996. `doi:10.1006/inco.1996.0008`.

17 Bartek Klin. Bialgebras for structural operational semantics: An introduction. *Theor. Comput. Sci.*, 412(38):5043–5069, 2011. `doi:10.1016/j.tcs.2011.03.023`.

18 Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980. `doi:10.1007/3-540-10235-3`.

19 Robin Milner. *Communication and concurrency*. PHI Series in computer science. Prentice Hall, 1989.

20 James H. Morris. *Lambda-Calculus Models of Programming Languages*. PhD thesis, Massachusetts Institute of Technology, 1968.

21 C.-H. L. Ong. *Correspondence between Operational and Denotational Semantics: The Full Abstraction Problem for PCF*, page 269?356. Oxford University Press, Inc., USA, 1995.

22 Andrew M. Pitts. Reasoning about local variables with operationally-based logical relations. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science, New Brunswick, New Jersey, USA, July 27-30, 1996*, pages 152–163. IEEE Computer Society, 1996. `doi:10.1109/LICS.1996.561314`.

23 Andrew M. Pitts. A note on logical relations between semantics and syntax. *Log. J. IGPL*, 5(4):589–601, 1997. `doi:10.1093/jigpal/5.4.589`.

24 Andrew M. Pitts. Parametric polymorphism and operational equivalence. *Math. Struct. Comput. Sci.*, 10(3):321–359, 2000. URL: `http://journals.cambridge.org/action/displayAbstract?aid=44651`.

25 Andrew M. Pitts. Typed operational reasoning. In Benjamin C. Pierce, editor, *Advanced Topics in Types and Programming Languages*, chapter 7. The MIT Press, 2004.

26 Gordon D. Plotkin. A structural approach to operational semantics. *J. Log. Algebr. Program.*, 60-61:17–139, 2004.

27 Andrei Popescu. Weak bisimilarity coalgebraically. In *Algebra and Coalgebra in Computer Science, Third International Conference, CALCO 2009, Udine, Italy, September 7–10, 2009. Proceedings*, pages 157–172, 2009. `doi:10.1007/978-3-642-03741-2_12`.

28 Damien Pous and Davide Sangiorgi. Enhancements of the bisimulation proof method. In Davide Sangiorgi and Jan J. M. M. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, volume 52 of *Cambridge tracts in theoretical computer science*, pages 233–289. Cambridge University Press, 2012.

29 Jan Rothe and Dragan Masulovic. Towards weak bisimulation for coalgebras. *Electr. Notes Theor. Comput. Sci.*, 68(1):32–46, 2002. `doi:10.1016/S1571-0661(04)80499-7`.

30 Jan J. M. M. Rutten. A note on coinduction and weak bisimilarity for while programs. *ITA*, 33(4/5):393–400, 1999. `doi:10.1051/ita:1999125`.

31 Davide Sangiorgi and Robin Milner. The problem of "weak bisimulation up to". In Rance Cleaveland, editor, *CONCUR '92, Third International Conference on Concurrency Theory, Stony Brook, NY, USA, August 24–27, 1992, Proceedings*, volume 630 of *Lecture Notes in Computer Science*, pages 32–46. Springer, 1992. `doi:10.1007/BFb0084781`.

32 Sam Staton. Relating coalgebraic notions of bisimulation. *Log. Methods Comput. Sci.*, 7(1), 2011. `doi:10.2168/LMCS-7(1:13)2011`.

33 Stelios Tsampas, Andreas Nuyts, Dominique Devriese, and Frank Piessens. A categorical approach to secure compilation. In Daniela Petrisan and Jurriaan Rot, editors, *Coalgebraic Methods in Computer Science – 15th IFIP WG 1.3 International Workshop, CMCS 2020, Colocated with ETAPS 2020, Dublin, Ireland, April 25-26, 2020, Proceedings*, volume 12094 of *Lecture Notes in Computer Science*, pages 155–179. Springer, 2020. `doi:10.1007/978-3-030-57201-3_9`.

34 Daniele Turi. Categorical modelling of structural operational rules: Case studies. In *Category Theory and Computer Science, 7th International Conference, CTCS '97, Santa Margherita Ligure, Italy, September 4–6, 1997, Proceedings*, pages 127–146, 1997. `doi:10.1007/BFb0026985`.

35 Daniele Turi and Gordon D. Plotkin. Towards a mathematical operational semantics. In *Proceedings, 12th Annual IEEE Symposium on Logic in Computer Science, Warsaw, Poland, June 29 – July 2, 1997*, pages 280–291, 1997. `doi:10.1109/LICS.1997.614955`.

36 Rob J. van Glabbeek. Full abstraction in structural operational semantics (extended abstract). In Maurice Nivat, Charles Rattray, Teodor Rus, and Giuseppe Scollo, editors, *Algebraic Methodology and Software Technology (AMAST '93), Proceedings of the Third International Conference on Methodology and Software Technology, University of Twente, Enschede, The Netherlands, 21–25 June, 1993*, Workshops in Computing, pages 75–82. Springer, 1993.

37 Rob J. van Glabbeek. On cool congruence formats for weak bisimulations. *Theor. Comput. Sci.*, 412(28):3283–3302, 2011. `doi:10.1016/j.tcs.2011.02.036`.

38 Mitchell Wand. Fixed-point constructions in order-enriched categories. *Theor. Comput. Sci.*, 8:13–30, 1979. `doi:10.1016/0304-3975(79)90053-7`.

39 Glynn Winskel and Mogens Nielsen. Models for concurrency. *DAIMI Report Series*, 22(463), November 1993. `doi:10.7146/dpb.v22i463.6936`.

A The simply WB cool rule format

In this section we introduce a few notions relevant to the simply WB cool rule format taken from [37]. For some n-ary operator o in a language $\mathcal{L}$, the *rules of* o are all the rules with source $o(x_1, \ldots, x_n)$. The following characterize GSOS rules based on the shape(s) of its premise(s) and the shape of its conclusion.

- An operator in $\mathcal{L}$ is *straight* if it has no rules in which a variable occurs multiple times in the left-hand side of its premises.
- An operator is *smooth* if it is straight and has no rules in which a variable occurs both in the target and in the left-hand side of a premise.
- An argument $i \in \mathbb{N}$ of an operator o is *active* if there is a rule of o in which x_i is the left-hand side of a premise.
- A variable x occurring in a term t is *receiving in* t if t is the target of a rule in $\mathcal{L}$ in which x is the right-hand side of a premise. An argument $i \in \mathbb{N}$ of an operator o is receiving if a variable x is receiving in a term t that has a subterm $o(v_1, \ldots, v_n)$ with x occurring in v_i.
- A rule of the form of $\dfrac{x_i \xrightarrow{\tau} y}{o(x_1, \ldots, x_n) \xrightarrow{\tau} o(x_1, \ldots, x_n)[y/x_i]}$ for $1 \leq i \leq n$ is called a patience rule for the ith argument of o, where $t[y/x]$ stands for term t with all occurrences of x replaced by y.

We can now give the complete definition of the simply WB cool rule format.

▶ **Definition A.1.** *A GSOS language $\mathcal{L}$ is simply WB cool if it is positive and the following conditions are all true.*

1. *All operators in $\mathcal{L}$ are straight.*
2. *The only rules in $\mathcal{L}$ with τ-premises are patience rules.*
3. *Every active argument of an operator has a patience rule.*
4. *Every receiving argument of an operator has a patience rule.*
5. *All operators in $\mathcal{L}$ are smooth.*

B Selected proofs

In this section of the appendix we include the proofs of Proposition 3.1 and Theorem 3.16.

B.1 Equivalence of the two representations of the continuity criterion

Proof of Proposition 3.1. We introduce the friendlier notation $\overline{\Sigma}f$ in place of $\lambda \circ \Sigma^*\langle 1, f\rangle$ for any $f : X \to TX$. (1) $\implies$ (2) is immediate since $\rho = \lambda \circ \theta$, where $\theta : \Sigma \implies \Sigma^*$ is the universal natural transformation sending Σ to its free monad. For (2) $\implies$ (1), we first note that for each "link" f_i, $\overline{\Sigma}f_i$ is (equivalently) defined via "structural recursion with accumulators" (see [35, Theorem 5.1]), i.e. it is the unique morphism making the following diagram commute.

$$\begin{tikzcd}
\Sigma\Sigma^*X \arrow[rr, "\mu\circ\theta_{\Sigma^*}"] \arrow[d, "\Sigma\langle 1,\overline{\Sigma}f_i\rangle"] & & \Sigma^*X \arrow[d, dashed, "\overline{\Sigma}f_i"', "\exists!"] & X \arrow[l, "\eta"'] \arrow[dl, "T\eta\circ f_i"] \\
\Sigma(\mathrm{Id}\times T)\Sigma^*X \arrow[r, "\rho_{\Sigma^*}"] & T\Sigma^*\Sigma^*X \arrow[r, "T\mu"] & T\Sigma^*X &
\end{tikzcd}$$

This makes $\overline{\Sigma}f_i$ the (necessarily unique) *homomorphic extension* of $T\mu\circ\rho_{\Sigma^*}$ along $T\eta\circ f_i$. Continuity of composition in $\mathcal{K}\ell(T)$ gives

$$T\eta\circ\bigvee_i f_i = \bigvee_i (T\eta\circ f_i) \tag{13}$$

$T\eta\circ f_i = \overline{\Sigma}f_i\circ\eta$, thus $\bigvee_i(\overline{\Sigma}f_i\circ\eta)$ exists and also

$$\bigvee_i(\overline{\Sigma}f_i\circ\eta) = (\bigvee_i\overline{\Sigma}f_i)\circ\eta \tag{14}$$

Via similar reasoning, we have

$$T\mu\circ\bigvee_i(\rho_{\Sigma^*}\circ\Sigma\langle 1,\overline{\Sigma}f_i\rangle) = \bigvee_i(T\mu\circ\rho_{\Sigma^*}\circ\Sigma\langle 1,\overline{\Sigma}f_i\rangle) \tag{15}$$

$$\bigvee_i(\overline{\Sigma}f_i\circ\mu\circ\theta_{\Sigma^*}) = (\bigvee_i\overline{\Sigma}f_i)\circ\mu\circ\theta_{\Sigma^*} \tag{16}$$

Equation (2) allows us to rewrite Equation (15) as

$$T\mu\circ\rho_{\Sigma^*}\circ\Sigma\langle 1,\bigvee_i\overline{\Sigma}f_i\rangle = \bigvee_i(T\mu\circ\rho_{\Sigma^*}\circ\Sigma\langle 1,\overline{\Sigma}f_i\rangle) \tag{17}$$

By taking the supremum over i of the i-dependent arrows in the previous diagram, Equation (13), (14), (16) and (17) allow us to present $\bigvee_i \overline{\Sigma} f_i$ as a morphism that makes the following diagram commute.

$$\begin{tikzcd}
\Sigma\Sigma^*X \arrow[rr, "\mu\circ\theta_{\Sigma^*}"] \arrow[d, "\Sigma\langle 1,\bigvee_i \overline{\Sigma} f_i\rangle"] & & \Sigma^*X \arrow[d, "\bigvee_i \overline{\Sigma} f_i"'] & X \arrow[l, "\eta"'] \arrow[dl, "T\eta\circ\bigvee_i f_i"] \\
\Sigma(\mathrm{Id}\times T)\Sigma^*X \arrow[r, "\rho_{\Sigma^*}"] & T\Sigma^*\Sigma^*X \arrow[r, "T\mu"] & T\Sigma^*X &
\end{tikzcd}$$

But there can only be one such morphism, namely $\overline{\Sigma}(\bigvee_i f_i)$, the unique homomorphic extension of $T\mu \circ \rho_{\Sigma^*}$ along $T\eta \circ \bigvee_i f_i$. In other words, $\bigvee_i \overline{\Sigma} f_i = \overline{\Sigma}(\bigvee_i f_i)$. ◀

B.2 Correspondence with the simply WB cool rule format

Proof of Theorem 3.16. In order to prove that any language $\mathcal{L}$ in the simply WB cool format automatically satisfies the three criteria, it suffices to show that (the GSOS law induced by) any arbitrary rule in $\mathcal{L}$ satisfies them. First of all, we know that rules in the simply WB cool format are of the form $\dfrac{\{x_i \xrightarrow{c_i} y_i \mid i \in I\}}{o(x_1,\ldots,x_n) \xrightarrow{\alpha} t}$ for $I \subseteq \{1,\ldots,n\}$[37, §3], or simply $\dfrac{\{x_i \xrightarrow{c_i} y_i \mid i \in I\}}{o(\overrightarrow{x}) \xrightarrow{\alpha} t}$, where o is an n-ary operator. We thus consider an arbitrary rule in the above form and proceed by distinguishing by the number of active arguments.

B.2.1 0 active arguments

All cases are trivial.

B.2.2 1 active argument

The rule is of the form $\dfrac{x_j \xrightarrow{c} y}{o(\overrightarrow{x}) \xrightarrow{\alpha} t}$ meaning that the rule is active on a position j. There is only a single premise because of the first requirement (straightness) in Definition A.1.

B.2.2.1 Patience rule

If $c = \tau$ then by the second requirement of Definition A.1 this rule has to be a patience rule of the form $\dfrac{x_j \xrightarrow{\tau} y}{o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})[y/x_j]}$.

Criterion 1.

$$\bigvee_i \rho(o(\overrightarrow{x}, f_i(\overrightarrow{x}))) = \bigvee_i \{\tau, o(\overrightarrow{x})[y/x_j] \mid (\tau, y_i) \in f_i(x_j)\} =$$
$$\{\tau, o(\overrightarrow{x})[y/x_j] \mid (\tau, y_i) \in \bigvee_i (f_i(x_j))\} = \rho(o(\overrightarrow{x}, \bigvee_i (f_i(\overrightarrow{x})))$$

Criterion 2. $\rho(o(\overrightarrow{x}, (\eta \vee f)(\overrightarrow{x})))$ induces a single transition $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})$ for all f, which is included in $\eta_x \circ \theta_X$.

Criterion 3. The only transitions in $\rho(o(\overrightarrow{x}, (f \diamond f)(\overrightarrow{x})))$ are $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})[z/x_j]$ when $x_j \xrightarrow{\tau} y \xrightarrow{\tau} z$, i.e. $(\tau, y) \in f(x_j), (\tau, z) \in g(y)$ for some y, z. Running $(\lambda_X \circ \Sigma^*\langle 1,f\rangle) \diamond (\rho_X \circ \Sigma\langle 1,f\rangle)$ [2], which is the second iteration on the right side, we can see that $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})[y/x_j] \xrightarrow{\tau} o(\overrightarrow{x})[z/y]$, which satisfies the criterion.

B.2.2.2 Impatient rule

This time we have $c \neq \tau$ which entails, by the third requirement of Definition A.1, the presence of a patience rule for argument j.

Criterion 1. Similar to B.2.2.1.

$$\bigvee_i \rho(o(\overrightarrow{x}, f_i(\overrightarrow{x}))) = \bigvee_i \{c, o(\overrightarrow{x})[y/x_j] \mid (c, y_i) \in f_i(x_j)\} =$$
$$\{c, o(\overrightarrow{x})[y/x_j] \mid (c, y_i) \in \bigvee_i (f_i(x_j))\} = \rho(o(\overrightarrow{x}, \bigvee_i (f_i(\overrightarrow{x})))$$

Criterion 2. Similarly to B.2.2.1, the presence of the patience rule for argument j means that there is always transition $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})$ for all f on the left side, which is included on the right side by $\eta_x \circ \theta_X$. The transitions induced by f exist on both sides.

Criterion 3. Transitions on the left side occur if and only if there are w, z such that $x_j \xrightarrow{\tau} w \xrightarrow{c} z$ or $x_j \xrightarrow{c} w \xrightarrow{\tau} z$, i.e. $(\tau, w) \in f(x_j), (c, z) \in f(w)$ or $(c, w) \in f(x_j), (\tau, z) \in f(w)$. We also have to distinguish between y in premise $x_j \xrightarrow{c} y$ being receiving in t or not. For each case, we give the transition(s) on the left side (of Criterion 3) and the respective iteration step where the left-side transitions appear on the right side (of Criterion 3).

1. y is not receiving, $x_j \xrightarrow{\tau} w \xrightarrow{c} z$.
   - $\rho_X \circ \Sigma\langle 1,f \diamond f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t$
   - $(\lambda_X \circ \Sigma^*\langle 1,f\rangle) \diamond (\rho_X \circ \Sigma\langle 1,f\rangle)$: $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})[w/x_j] \xrightarrow{\alpha} t$
2. y is not receiving, $x_j \xrightarrow{c} w \xrightarrow{\tau} z$.
   - $\rho_X \circ \Sigma\langle 1,f \diamond f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t$
   - $\rho_X \circ \Sigma\langle 1,f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t$
3. y is receiving, $x_j \xrightarrow{\tau} w \xrightarrow{c} z$.
   - $\rho_X \circ \Sigma\langle 1,f \diamond f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t(z)$
   - $(\lambda_X \circ \Sigma^*\langle 1,f\rangle) \diamond (\rho_X \circ \Sigma\langle 1,f\rangle)$: $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})[w/x_j] \xrightarrow{\alpha} t(z)$
4. y is receiving, $x_j \xrightarrow{c} w \xrightarrow{\tau} z$

   This is the trickiest case. Let us look back at the rule in question (with y receiving), which is $\dfrac{x_j \xrightarrow{c} y}{o(\overrightarrow{x}) \xrightarrow{\alpha} t(y)}$. The key observation is that requirement 4 in Definition A.1, requiring that every receiving argument of an operator has a patience rule, implies that no matter how complex the receiving expression $t(y)$ is, there will be patience rules in place to ensure a derivation amounting to $\dfrac{x \xrightarrow{\tau} y}{s(x) \xrightarrow{\tau} s(y)}$ for each sub-expression $s(y)$ in $t(y)$ that "receives" y. The number of iterations on the right-hand required in order to trigger all necessary patience rules depends on the number of sub-expressions s.

   For example, let $t(y) \triangleq d(e(l(y, 0)), e(l(0, y)))$, where d, l are binary operations, e is a unary operation and 0 is some term. Due to requirement 4, d, l, e will all have patience

[2] Recall that $(\lambda_X \circ \Sigma^*\langle 1,f\rangle) \circ \theta = \rho_X \circ \Sigma\langle 1,f\rangle$.

rules in all positions and we need exactly two iterations to trigger the two patience rules in each of the positions of d. More generally,

- $\rho_X \circ \Sigma\langle 1,f \diamond f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t(z)$
- $(\lambda_X \circ \Sigma^*\langle 1,f\rangle)^* \diamond (\rho_X \circ \Sigma\langle 1,f\rangle)$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t(w) \xrightarrow{\tau^*} t(z)$.

B.2.3 2 or more active arguments

We first note that the only rules with τ-premises are patience rules, which are already covered in B.2.2.1 and so we move on to $c_1, c_2 \neq \tau$ with the rule being of the form of $\dfrac{x_i \xrightarrow{c_1} y_1 \quad x_j \xrightarrow{c_2} y_2}{o(\overrightarrow{x}) \xrightarrow{\alpha} t}$. The third requirement of Definition A.1 means that there are patience rules for arguments i and j in o.

Criterion 1. Similar to the case for Criterion 1 in B.2.2.1.

Criterion 2. Similar to the case for Criterion 2 in B.2.2.1.

Criterion 3. There are four separate cases where a transition occurs in $\rho(o(\overrightarrow{x}, (f \diamond f)(\overrightarrow{x})))$, as a c_1-transition and a c_2-transition may occur in either step for both subterms.

1. $x_i \xrightarrow{\tau} w_1 \xrightarrow{c_1} z_1$ and $x_j \xrightarrow{\tau} w_2 \xrightarrow{c_2} z_2$
2. $x_i \xrightarrow{\tau} w_1 \xrightarrow{c_1} z_1$ and $x_j \xrightarrow{c_2} w_2 \xrightarrow{\tau} z_2$
3. $x_i \xrightarrow{c_1} w_1 \xrightarrow{\tau} z_1$ and $x_j \xrightarrow{c_2} w_2 \xrightarrow{\tau} z_2$
4. $x_i \xrightarrow{c_1} w_1 \xrightarrow{\tau} z_1$ and $x_j \xrightarrow{\tau} w_2 \xrightarrow{c_2} z_2$.

In addition, y_1 and y_2 can each be either receiving or not receiving in t and, as this affects transitions the same way as y being receiving in B.2.2.2.

1. $x_i \xrightarrow{\tau} w_1 \xrightarrow{c_1} z_1$ and $x_j \xrightarrow{\tau} w_2 \xrightarrow{c_2} z_2$
   Here, whether y_1 and y_2 are receiving or not does not make a difference and we write $t(y_1, y_2)$ to denote a term which potentially has instances of y_1 and y_2.
   - $\rho_X \circ \Sigma\langle 1,f \diamond f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t(z_1, z_2)$
   - $(\lambda_X \circ \Sigma^*\langle 1,f\rangle) \diamond (\lambda_X \circ \Sigma^*\langle 1,f\rangle) \diamond (\rho_X \circ \Sigma\langle 1,f\rangle)$:
     $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})[w_1/x_i] \xrightarrow{\tau} o(\overrightarrow{x})[w_1/x_i][w_2/x_j] \xrightarrow{\alpha} t(z_1, z_2)$

   We can see that we need to iterate three times on the right-hand side: one to trigger the patience rule on position i, one to trigger the patience rule on position j on the new term and one more to trigger the main rule.
2. $x_i \xrightarrow{\tau} w_1 \xrightarrow{c_1} z_1$ and $x_j \xrightarrow{c_2} w_2 \xrightarrow{\tau} z_2$
   In this case y_2 being receiving makes a difference, while y_1 does not. Let us first deal with the non-receiving case for y_2.
   - $\rho_X \circ \Sigma\langle 1,f \diamond f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t(z_1)$
   - $(\lambda_X \circ \Sigma^*\langle 1,f\rangle) \diamond (\rho_X \circ \Sigma\langle 1,f\rangle)$: $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})[w_1/x_i] \xrightarrow{\alpha} t(z_1)$

   The first iteration will trigger the patience rule for i, while the second produces the α-transition. Variable y_2 is not receiving and so nothing else is needed. On the other hand, if y_2 is receiving, we see that
   - $\rho_X \circ \Sigma\langle 1,f \diamond f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t(z_1, z_2)$
   - $(\lambda_X \circ \Sigma^*\langle 1,f\rangle)^* \diamond (\lambda_X \circ \Sigma^*\langle 1,f\rangle) \diamond (\rho_X \circ \Sigma\langle 1,f\rangle)$:
     $o(\overrightarrow{x}) \xrightarrow{\tau} o(\overrightarrow{x})[w_1/x_i] \xrightarrow{\alpha} t(z_1, w_2) \xrightarrow{\tau^*} t(z_1, z_2)$

   Similarly to the fourth case of Criterion 3 in B.2.2.2, requirement 4 in Definition A.1 guarantees that there will be a sequence of patience rules that gives $t(z_1, w_2) \xrightarrow{\tau^*} t(z_1, z_2)$.

3. $x_i \xrightarrow{c_1} w_1 \xrightarrow{\tau} z_1$ and $x_j \xrightarrow{c_2} w_2 \xrightarrow{\tau} z_2$
   If y_1 and y_2 are not receiving, this is very similar to the first case. If y_1 and/or y_2 are receiving, then requirement 4 in Definition A.1 comes into play in the same manner as before. For instance, if both y_1 and y_2 are receiving:
   - $\rho_X \circ \Sigma\langle 1, f \diamond f\rangle$: $o(\overrightarrow{x}) \xrightarrow{\alpha} t(z_1, z_2)$
   - $(\lambda_X \circ \Sigma^*\langle 1, f\rangle)^* \diamond (\lambda_X \circ \Sigma^*\langle 1, f\rangle)^* \diamond (\rho_X \circ \Sigma\langle 1, f\rangle)$:
     $o(\overrightarrow{x}) \xrightarrow{\alpha} t(w_1, w_2) \xrightarrow{\tau^*} t(z_1, w_2) \xrightarrow{\tau^*} t(z_1, z_2)$
4. $x_i \xrightarrow{c_1} w_1 \xrightarrow{\tau} z_1$ and $x_j \xrightarrow{\tau} w_2 \xrightarrow{c_2} z_2$
   Very similar to the second case.

In the presence of three or more active arguments we can apply the same principles, the only difference being that the maximum number of necessary iterations on the right side will be higher, as more patience rules will have to be triggered. ◀