Compositionality in Coalgebraic Trace Semantics

Robin Jourde
Univ. Savoie Mont Blanc, CNRS, LAMA, Chambéry, France

Henning Urbat
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Sergey Goncharov
University of Birmingham, UK

Stelios Tsampas
University of Southern Denmark, Odense, Denmark

Jonas Forster
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

Abstract

A key requirement on any well-behaved process language is its *compositionality*: behavioural equivalence of processes should be respected by the constructors of the language. Turi and Plotkin's *abstract GSOS* provides an elegant bialgebraic framework for modelling rule formats that guarantee compositionality from the outset. Their original results, however, are restricted to compositionality of strong bisimilarity, a rather fine-grained notion of process equivalence. In the present paper, we demonstrate that Turi and Plotkin's approach also applies to *trace equivalence*, which only observes external actions of processes. To this end, we revisit the general compositionality result of their original theory and present it in a refined form with regard to the required naturality conditions. This step makes abstract GSOS applicable over Kleisli categories and thereby enables reasoning about compositionality in the setting of *coalgebraic trace semantics*. As our main contribution, we introduce *De Simone laws*, a type of GSOS laws over Kleisli categories, and prove that their operational models are compositional for coalgebraic trace equivalence. This result recovers and explains compositionality of the well-known *De Simone* rule format for labelled transition systems in a natural categorical setting. As a further application, we derive from our general framework a novel De Simone-type format for *probabilistic* systems, compositional for probabilistic trace equivalence.

2012 ACM Subject Classification Theory of computation → Process calculi

Keywords and phrases Coalgebra, Operational Semantics, Process Algebra, Abstract GSOS, Trace Semantics, Rule Formats

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2026.38

Related Version *Full Version*: `https://arxiv.org/abs/2605.18285`

Funding *Henning Urbat*: Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 569130867.
Sergey Goncharov: Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 527481841.
Stelios Tsampas: Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 527481841.
Jonas Forster: Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 434050016.

1 Introduction

One of the primary challenges in concurrency theory is the development of effective tools and techniques to reason about *equivalence* of processes. Numerous notions of process equivalence at different levels of granularity have been studied. Among the most prominent ones, lying

at opposite ends of the linear-time branching-time spectrum [46], are *strong bisimilarity*, corresponding to a low-level observer with full access to internal states and transitions of the given processes, and *trace equivalence*, where only external actions are observed. Regardless of the chosen flavour of equivalence, a key desideratum is that it forms a *congruence*, which means that it is respected by all process constructors. For instance, one would expect two processes $p \parallel q$ and $p' \parallel q$ to be equivalent if their subprocesses p and p' are. The importance of this property is that it enables *compositional* reasoning: it ensures that the observable behaviour of a complex process is fully determined by that of each individual component.

However, actually *proving* that a process language is compositional for a given type of equivalence is cumbersome, in particular in advanced settings with higher-order features, computational effects, etc. Hence, there has been longstanding interest in syntactic rule formats that guarantee compositionality for all languages whose specification adheres to the given format [4]. Well-known examples include the *GSOS* [11] and *tyft/tyxt* formats [24], which are compositional for strong bisimilarity, and the *De Simone* format [17, 10], a fragment of GSOS compositional for trace semantics. Generally, coarse-grained process equivalences with very limited internal access to processes, such as trace equivalence, tend to require severe syntactic restrictions on the operational rules to ensure compositionality. Standardly, those restrictions are discovered ad hoc, as are the proofs of their congruence properties.

In the present paper, we develop a systematic *categorical* approach to congruence formats for trace equivalence that treats the given process language and the structural rules specifying its operational semantics as abstract parameters. Our work draws from two different theories:

Coalgebraic Trace Semantics. Labelled transition systems and other types of state-based structures that underlie the operational semantics of process languages admit a uniform categorical abstraction in the form of *coalgebras* [40] for a functor encapsulating the system type. Hasuo et al. [25] have shown that various notions of trace semantics can be captured at coalgebraic generality by considering final coalgebras over order-enriched *Kleisli categories*.

Abstract GSOS. Turi and Plotkin [44] have proposed an elegant categorical approach to GSOS-type rule formats. The idea is to present GSOS rules as *GSOS laws*, a type of natural transformation that distributes the *syntax* of a language over its *behaviour*, both modelled by functors. The core selling point of their framework is its free compositionality theorem: for every language modelled by a GSOS law, behavioural equivalence (i.e. equality in the final coalgebra of the behaviour functor) is a congruence. The compositionality of GSOS rules w.r.t. strong bisimilarity [11] is one of many instances [43, 30] of this general result.

In the light of the above work, we base our abstract theory of congruence for traces on a (seemingly) simple strategy: run the machinery of abstract GSOS over Kleisli categories, the setting for coalgebraic trace semantics. As a first proof of concept, we demonstrate that just like GSOS rules are captured by GSOS laws over the category of sets, De Simone rules correspond to *De Simone laws*, a novel type of GSOS laws living in the Kleisli category of an additive monad. Our key insight is that the ad hoc syntactic restrictions imposed on De Simone rules (regarding affineness of terms) are precisely reflected by certain naturality conditions on their associated De Simone laws. At this point, a technical challenge arises: Since De Simone laws are only natural w.r.t. *some* Kleisli morphisms, Turi and Plotkin's theory of congruence (which is tailored to fully natural GSOS laws) does not immediately apply to this setting. Therefore, we revisit Turi and Plotkin's original congruence theorem for abstract GSOS and present it in a refined form that analyses the precise required naturality conditions (Theorem 4.5). Since the original proof [44] heavily relies on naturality, the refined congruence result requires a conceptually different approach beyond mere proof inspection.

The refined congruence theorem for abstract GSOS allows us to establish our main result (Theorem 5.15): coalgebras specified by De Simone laws admit compositional trace semantics. Besides incorporating the congruence property for classical De Simone rules [10] into the abstract GSOS framework, this result gives rise to novel congruence formats for *effectful* settings, thus highlighting the scope and effectiveness of our theory. As an advanced case study, we derive a De Simone-type format for (almost-surely terminating) *probabilistic transition systems*. While several congruence formats for probabilistic or weighted bisimilarity have emerged in the literature [8, 32, 14], our probabilistic De Simone format is (to the best of our knowledge) the first congruence format for probabilistic trace semantics.

Related Work. Since the introduction of abstract GSOS [44], a categorical understanding of congruence formats for equivalence notions coarser than strong bisimilarity has been pursued. Most previous approaches in this direction conceptually depart from Turi and Plotkin's work (rather than using it) and develop the respective theory of congruence largely from scratch.

Klin [28, 29] presents an abstract approach to compositional semantics parametric in a *coalgebraic logic*, a form of dual adjunction that supplies the targeted notion of semantics. Technically, this work is not compatible with ours, although some instances (such as compositionality of standard De Simone rules) are shared. Klin's logical framework is highly flexible and expressive. One drawback is that its application requires, on top of the coalgebraic logic, coming up with a further technical ingredient in the form of a *logical distributive law*, which is typically not trivial to find and subject to a complex compatibility condition. This contrasts our approach where only the standard data of coalgebraic trace semantics is needed to instantiate the theory, and the technical conditions of the congruence result amount to naturality conditions over a Kleisli category with a simple underlying intuition.

The interaction of abstract GSOS with monadic, or generally effectful, behaviour has been a topic of interest for a long time [3, 9, 23, 21, 38, 5, 45], the common goal typically being compositionality for various effectful program equivalences. Notably, Abou-Saleh and Pattinson develop a form of abstract GSOS over a *mixed* Kleisli setting [2, 1], and study the semantics given by final coalgebras in a Kleisli category. They propose a (rather technical and hard to verify) *condition on cones* to guarantee compositionality. Unlike our own Kleisli-based approach, their theory of congruence is not based on Turi and Plotkin's results. Moreover, in terms of applications their work is primarily aimed at reasoning about stateful imperative programming languages and does not encompass De Simone-type specifications.

Finally, Tsampas et al. [42] consider GSOS laws featuring behaviours of the form TB, where T is a monad whose Kleisli category is order-enriched in a manner that TB-coalgebras can be *saturated* [13]. Their theory of congruence applies to weak bisimilarity rather than trace semantics and also departs from the standard theory of Turi and Plotkin.

2 Categorical Preliminaries

Our theory of compositional trace semantics rests on various abstractions using the language of category theory [37], notably (co)algebras and Kleisli extensions reviewed below. Readers are assumed to be familiar with functors, natural transformations, (co)products, and monads.

Notation. Given objects X_1, X_2 in a category $\mathcal{C}$, we write $X_1 \times X_2$ for their product, $\mathsf{outl}\colon X_1 \times X_2 \to X_1$ and $\mathsf{outr}\colon X_1 \times X_2 \to X_2$ for the projections, and $\langle f_1, f_2\rangle\colon Y \to X_1 \times X_2$ for the pairing of morphisms $f_i\colon Y \to X_i$, $i = 1, 2$. We write 1 for the terminal object and $!\colon X \to 1$ for the unique morphism. Dually, $X_1 + X_2$ denotes the coproduct, $\mathsf{inl}\colon X_1 \to X_1 + X_2$

and $\mathsf{inr}\colon X_2 \to X_1 + X_2$ the injections, and $[f_1, f_2]\colon X_1 + X_2 \to Y$ a co-pairing. In the category **Set** of sets and functions, products and coproducts are cartesian products and disjoint unions, resp., and $1 = \{*\}$. Given a monad T, we write η^T and μ^T for the unit and multiplication; the superscript T is usually dropped. A *pre-natural transformation* between functors $F, G\colon \mathcal{C} \to \mathcal{D}$ is a family of morphisms $\alpha_X\colon FX \to GX$ $(X \in \mathcal{C})$. It is *natural w.r.t. a morphism* $f\colon X \to Y$ of $\mathcal{C}$ if $\alpha_Y \circ Ff = Gf \circ \alpha_X$. A natural transformation is thus a pre-natural transformation that is natural w.r.t. all morphisms. We write α for $\alpha_X, \alpha_Y, \ldots$

Algebras. Algebraic structures admit a categorical abstraction in the form of algebras for an endofunctor F on a category $\mathcal{C}$. An *F-algebra* is a pair (A, a) of an object A and a morphism $a\colon FA \to A$. A *morphism* from (A, a) to an F-algebra (B, b) is a morphism $h\colon A \to B$ of $\mathcal{C}$ with $h \circ a = b \circ Fh$. Algebras for F and their morphisms form a category, and an *initial* F-algebra is simply an initial object in that category. We denote the initial algebra by $(\mu F, \iota)$ if it exists. By Lambek's Lemma [34], its structure $\iota\colon F(\mu F) \to \mu F$ is an isomorphism.

A *free F-algebra* on an object X of $\mathcal{C}$ is an F-algebra (F^*X, ι_X) together with a morphism $\eta_X\colon X \to F^*X$ of $\mathcal{C}$ such that for every algebra (A, a) and every morphism $h\colon X \to A$ of $\mathcal{C}$, there is a unique F-algebra morphism $h^*\colon (F^*X, \iota_X) \to (A, a)$ with $h = h^* \circ \eta_X$. If $\mathcal{C}$ has an initial object 0, then $F^*0 = \mu F$. If free algebras exist on every object, their formation induces a monad $F^*\colon \mathcal{C} \to \mathcal{C}$, the *free monad* generated by F [7]. Every F-algebra (A, a) yields an Eilenberg-Moore algebra $\hat{a} := \mathsf{id}_A^*\colon F^*A \to A$, where $\mathsf{id}_A\colon A \to A$ is the identity.

▶ **Example 2.1** (Algebras for a Signature)**.** An (*algebraic*) *signature* consists of a set Σ of *operation symbols* and a map $\mathsf{ar}\colon \Sigma \to \mathbb{N}$ associating to every $\mathsf{f} \in \Sigma$ its *arity*. Every signature Σ induces an endofunctor on the category **Set**, denoted by the same letter Σ and defined by $\Sigma X = \coprod_{\mathsf{f}\in\Sigma} X^{\mathsf{ar}(\mathsf{f})}$. Endofunctors of this form are called *polynomial*. An algebra for the polynomial functor Σ is precisely an algebra for the signature Σ in the usual sense, that is, a set A equipped with an operation $\mathsf{f}^A\colon A^n \to A$ for every n-ary $\mathsf{f} \in \Sigma$. Morphisms of Σ-algebras are maps respecting the algebraic structure. Given a set X, the free algebra Σ^*X is carried by the set of Σ-terms with variables from X. The free algebra on the empty set is the initial algebra $\mu\Sigma$; it is formed by all *closed* Σ-terms. For every Σ-algebra (A, a), the Eilenberg-Moore algebra $\hat{a}\colon \Sigma^*A \to A$ evaluates terms in A.

Coalgebras. The dual concept of algebras are coalgebras, which form a categorical abstraction of state-based transition systems. A *coalgebra* for an endofunctor B on $\mathcal{C}$ is a pair (C, c) of an object C and a morphism $c\colon C \to BC$. A *morphism* from (C, c) to a B-coalgebra (D, d) is a morphism $h\colon C \to D$ of $\mathcal{C}$ such that $Bh \circ c = d \circ h$. Coalgebras for B and their morphisms form a category, and a *final* B-coalgebra, denoted $(\nu B, \tau)$, is a final object in that category.

▶ **Example 2.2.** Labelled transition systems (LTS) with or without explicit termination correspond, respectively, to coalgebras of type $c\colon C \to \mathcal{P}(L \times C + 1)$ and $c_0\colon C \to \mathcal{P}(L \times C)$, for a set L of labels. Ordinary LTS (i.e. without explicit termination) form a subclass of LTS with explicit termination by regarding all states as terminating: put $c(x) = c_0(x) \cup \{*\}$.

Kleisli Extensions. Given a monad T on a category $\mathcal{C}$, we write $\mathbf{Kl}(T)$ for its Kleisli category. Objects of $\mathbf{Kl}(T)$ are those of $\mathcal{C}$, and a morphism $f\colon X \to\!\!\!\circ\, Y$ of $\mathbf{Kl}(T)$ is a morphism $f\colon X \to TY$ of $\mathcal{C}$, with identities and composition induced by the monad structure. For example $\mathbf{Kl}(\mathcal{P})$ is the category of sets and relations. A *Kleisli extension* of an endofunctor $F\colon \mathcal{C} \to \mathcal{C}$ is an endofunctor $\bar{F}\colon \mathbf{Kl}(T) \to \mathbf{Kl}(T)$ making the diagram below commute.

$$\begin{array}{ccc} \mathbf{Kl}(T) & \xrightarrow{\bar{F}} & \mathbf{Kl}(T) \\ {\scriptstyle J_T}\uparrow & & \uparrow{\scriptstyle J_T} \\ \mathcal{C} & \xrightarrow{F} & \mathcal{C} \end{array}$$

Here J_T is the canonical left adjoint, acting on objects as the identity and on morphisms $f\colon X \to Y$ by $J_T f = \eta_Y \circ f$. Morphisms of $\mathbf{Kl}(T)$ of the form $J_T f$ are called *pure*. We usually put $J = J_T$ if the monad is clear from the context. Kleisli extensions of an endofunctor F are closely related to *distributive laws* of F over the monad T. The latter are natural transformations $\delta\colon FT \to TF$ subject to compatibility conditions [39]. Each distributive law $\delta\colon FT \to TF$ induces a Kleisli extension $\bar{F}\colon \mathbf{Kl}(T) \to \mathbf{Kl}(T)$, defined on objects as $\bar{F}X = FX$ and on morphisms $f\colon X \mathrel{-\!\!\circ\!\!\to} Y$ as $\bar{F}f = \delta_Y \circ Ff$. Conversely, each Kleisli extension of F arises from a unique distributive law. Thus there is a bijective correspondence between distributive laws and Kleisli extensions [39].

▶ **Proposition 2.3** ([25]). *Let $F\colon \mathcal{C} \to \mathcal{C}$ be an endofunctor with an initial algebra $\mu\Sigma$ and a free algebra F^*X on $X \in \mathcal{C}$, and let $\bar{F}\colon \mathbf{Kl}(T) \to \mathbf{Kl}(T)$ be a Kleisli extension. Then $\mu\bar{F} = \mu F$ and $\bar{F}^*X = F^*X$, with the $\bar{F}$-algebra structures and universal maps given by*

$$\bar{F}(\mu F) = F(\mu F) \xrightarrow{J\iota} \mu F, \qquad \bar{F}F^*X = FF^*X \xrightarrow{J\iota_X} F^*X, \qquad X \xrightarrow{J\eta_X} F^*X.$$

3 Coalgebraic Trace Semantics

We recall the Kleisli-based framework for coalgebraic trace semantics [25], following the streamlined presentation by Frank et al. [19] which simplifies some of the technical conditions of the original paper. The framework applies to coalgebras of type

$$c\colon C \to TBC \tag{3.1}$$

where B is an endofunctor (modelling input/output behaviour) and T is a monad (modelling a computational effect such as non-determinism or probabilities) on a category $\mathcal{C}$.

▶ **Example 3.1** (LTS). LTS with explicit termination are coalgebras of type (3.1) for $T = \mathcal{P}$ (the power set monad) and $BX = L \times X + 1$ on **Set**. We write $x \xrightarrow{a} y$ if $(a, y) \in c(x)$, and $x{\downarrow}$ if $* \in c(x)$. A *completed trace* at a state $x \in C$ is a string $a_1 \cdots a_n \in L^*$ such that $x = x_0 \xrightarrow{a_1} x_1 \xrightarrow{a_2} \cdots \xrightarrow{a_{n-1}} x_{n-1} \xrightarrow{a_n} x_n{\downarrow}$ for some $x_0, \dots, x_n \in C$. Let $\mathsf{tr}(x) \subseteq L^*$ be the set of completed traces at x. Note that L^* carries the initial algebra μB for the functor B. Thus completed trace semantics yields a morphism $\mathsf{tr}\colon C \mathrel{-\!\!\circ\!\!\to} \mu B$ in $\mathbf{Kl}(\mathcal{P})$. For an ordinary LTS (with all states terminating), the map tr sends a state x to the set of *partial traces* at x.

To generalize the trace map tr from LTS to coalgebras (3.1), some order-enrichment on the Kleisli category $\mathbf{Kl}(T)$ is required. A category $\mathcal{B}$ is *left strictly* $\mathbf{DCPO}_\perp$*-enriched* if each hom-set $\mathcal{B}(X, Y)$ carries the structure of a DCPO with $\perp$ such that composition is left-strict ($\perp \circ f = \perp$) and preserves directed joins in each component ($f \circ \bigsqcup_i g_i = \bigsqcup_i f \circ g_i$ and $(\bigsqcup_i g_i) \circ f = \bigsqcup_i g_i \circ f$). A functor $F\colon \mathcal{B} \to \mathcal{B}$ is *locally monotone* if $f \sqsubseteq g$ implies $Ff \sqsubseteq Fg$.

▶ **Definition 3.2.** A *setting for coalgebraic trace semantics* consists of a monad T and an endofunctor B on a category $\mathcal{C}$ with a Kleisli extension $\bar{B}\colon \mathbf{Kl}(T) \to \mathbf{Kl}(T)$ such that:

(1) the functor B has an initial algebra $(\mu B, \beta)$;
(2) the category $\mathbf{Kl}(T)$ is *left strictly* $\mathbf{DCPO}_\perp$*-enriched*;
(3) the functor $\bar{B}\colon \mathbf{Kl}(T) \to \mathbf{Kl}(T)$ is *locally monotone*.

The key property of any setting for coalgebraic trace semantics is the initial algebra/final coalgebra coincidence $\mu B = \mu\bar{B} = \nu\bar{B}$. The first equality is Proposition 2.3; the second one is the next proposition, which also appears (with stronger conditions on $\mathcal{C}$ and B) in [25].

▶ **Proposition 3.3** ([19]). *Let T, B, $\bar{B}$ be a setting for a coalgebraic trace semantics. Then the functor $\bar{B}\colon \mathbf{Kl}(T) \to \mathbf{Kl}(T)$ has a final coalgebra given by $\mu B \xrightarrow{J\beta^{-1}} B(\mu B) = \bar{B}(\mu B)$.*

(Recall that β is an isomorphism by Lambek's lemma.) To obtain coalgebraic trace semantics, note that a TB-coalgebra (3.1) in $\mathcal{C}$ is the same as a $\bar{B}$-coalgebra $c\colon C \to \bar{B}C$ in $\mathbf{Kl}(T)$. The *trace map* for c is the unique $\bar{B}$-coalgebra morphism to the final coalgebra $(\mu B, J\beta^{-1})$:

$$\mathsf{tr}_c\colon C \to \mu B. \tag{3.2}$$

We usually drop the subscript if c is clear from the context.

▶ **Example 3.4** (LTS [25]). LTS correspond to the setting for coalgebraic trace semantics where $T = \mathcal{P}$ and $BX = L \times X + 1$ (Example 3.1), and $\bar{B}$ is induced by the distributive law

$$\delta^B\colon L \times \mathcal{P}X + 1 \to \mathcal{P}(L \times X + 1), \qquad (a, S) \mapsto \{(a, s) \mid s \in S\}, \qquad * \mapsto \{*\}.$$

Then all the required conditions of Definition 3.2 are satisfied:

(1) The initial algebra for B is given by the set L^* with structure $\beta\colon L \times L^* + 1 \to L^*$, defined componentwise by $(a, w) \mapsto wa$, and $* \mapsto \varepsilon$. We let β append a to the right of a word w, which reflects how traces are formed in an LTS.

(2) $\mathbf{Kl}(\mathcal{P})$ is left strictly $\mathbf{DCPO}_\perp$-enriched w.r.t. the partial order given by pointwise inclusion ($f \sqsubseteq g\colon X \to Y$ iff $f(x) \subseteq g(x)$ for all $x \in X$).

(3) $\bar{B}$ is locally monotone as the law δ^B is monotone: $S \subseteq S'$ implies $\delta^B(a, S) \subseteq \delta^B(a, S')$.

The trace map from Example 3.1, corresponding to completed trace semantics of LTS, coincides with the coalgebraic trace map (3.2). Indeed, one readily verifies that completed trace semantics forms a $\bar{B}$-coalgebra morphism from (C, c) to the final coalgebra $(\mu B, J\beta^{-1})$.

4 Abstract GSOS, Revisited

Turi and Plotkin's [44] abstract GSOS models GSOS-style rule formats categorically and features a free congruence result. We recall their framework, and at the same time slightly refine it with respect to the required naturality conditions – a step that enables its application to coalgebraic trace semantics. Abstract GSOS is parametric in the following data:

- a base category $\mathcal{C}$ with binary products;
- an endofunctor Σ on $\mathcal{C}$ with an initial algebra $(\mu\Sigma, \iota)$ and free algebras $(\Sigma^* X, \iota_X)$ (so that Σ generates a free monad Σ^*);
- an endofunctor B on $\mathcal{C}$ with a final coalgebra $(\nu B, \tau)$.

Here, the functors Σ and B model the *syntax* and *behaviour* of a type of programs or processes under considerations, $\mu\Sigma$ is the object of closed process terms, and νB models abstract behaviours (e.g. tree unravellings or traces). The operational semantics of processes is captured by a (pre-)natural transformation that distributes syntax over behaviour:

▶ **Definition 4.1.** A *(pre-)GSOS law* of Σ over B is (pre-)natural transformation of type

$$\varrho_X \colon \Sigma(X \times BX) \to B\Sigma^* X \qquad (X \in \mathcal{C}). \tag{4.1}$$

The *operational model* of (4.1) is defined recursively [27, Prop. 2.4.6] as the unique coalgebra $\gamma \colon \mu\Sigma \to B(\mu\Sigma)$ such that the diagram below commutes.

$$\begin{array}{ccccc} \mu\Sigma & \xleftarrow{\quad\iota\quad} & & & \Sigma(\mu\Sigma) \\ {\scriptstyle\gamma}\downarrow & & & & \downarrow{\scriptstyle\Sigma\langle \mathsf{id},\gamma\rangle} \\ B(\mu\Sigma) & \xleftarrow{B\hat{\iota}} & B\Sigma^*(\mu\Sigma) & \xleftarrow{\varrho} & \Sigma(\mu\Sigma \times B(\mu\Sigma)) \end{array}$$

We denote the unique coalgebra morphism into the final coalgebra by $\mathsf{beh} \colon (\mu\Sigma, \gamma) \to (\nu B, \tau)$.

Intuitively, a pre-GSOS law ϱ encodes the operational rules of a given process language: for every constructor f, it specifies the one-step behaviour of a process $\mathsf{f}(p_1, \cdots, p_n)$, i.e. the Σ-terms it transitions into next, depending on the one-step behaviours of its subprocesses p_i. The operational model γ is the transition system that runs processes according to those rules, and the map beh sends every process to its abstract behaviour, e.g. its bisimilarity class or its set of traces, depending on the structure of the final coalgebra νB in the given setting.

▶ **Example 4.2.** Consider a process algebra with a parallel composition operator $(p, q) \mapsto p \parallel q$ specified by the operational rules (4.2), where a ranges over a fixed set L of action labels:

$$\frac{p \xrightarrow{a} p'}{p \parallel q \xrightarrow{a} p' \parallel q} \qquad \frac{q \xrightarrow{a} q'}{p \parallel q \xrightarrow{a} p \parallel q'} \tag{4.2}$$

$$\frac{x_{i_j} \xrightarrow{a_j} y_j \ (j \in J) \qquad x_{i_k} \overset{b_k}{\not\longrightarrow} \ (k \in K)}{\mathsf{f}(x_1, \ldots, x_n) \xrightarrow{a} t} \tag{4.3}$$

We take $\Sigma X = X \times X$, $BX = \mathcal{P}(L \times X)$ and devise a corresponding GSOS law from (4.2):

$$\varrho_X \colon (X \times \mathcal{P}(L \times X)) \times (X \times \mathcal{P}(L \times X)) \to \mathcal{P}(L \times \Sigma^* X) \qquad (X \in \mathbf{Set}),$$
$$\varrho_X(p, S, q, T) = \{(a, p' \| q) : (a, p') \in S\} \cup \{(a, p \| q') : (a, q') \in T\}.$$

Naturality of ϱ (w.r.t. functions, i.e. renamings of variables) amounts to the rules (4.2) being parametrically polymorphic, that is, they do not inspect the structure of the variables.

The above translation works more generally for *GSOS rules* [11], viz. rules of the form (4.3) where f is an n-ary operation symbol from a given signature Σ, $i_j, i_k \in \{1, \ldots, n\}$, the variables x_i and y_j are pairwise distinct, $a, a_j, b_k \in L$, and t is a Σ-term in the variables x_i and y_j. Turi and Plotkin [44] observed that every set $\mathcal{R}$ of GSOS rules can be presented as a GSOS law of the polynomial functor Σ over B. The operational model of the law is the LTS

$$\gamma \colon \mu\Sigma \to \mathcal{P}(L \times \mu\Sigma) \tag{4.4}$$

whose transitions are inductively determined by the given rules; that is, there is a transition $\mathsf{f}(t_1, \ldots, t_n) \xrightarrow{a} s$ iff there exists a rule (4.3) in $\mathcal{R}$ and terms $s_j \in \mu\Sigma$ $(j \in J)$ such that $t_{i_j} \xrightarrow{a_j} s_j$ for $j \in J$, $t_{i_k} \overset{b_k}{\not\longrightarrow}$ (i.e. there is no b_k-labelled transition from t_{i_k}) for $k \in K$, and the term s arises from t via the substitution $x_i \mapsto t_i$ and $y_j \mapsto s_j$.

In what follows, we gloss over the minor technical issue of final coalgebras not existing for functors involving the full power set functor $\mathcal{P}$; this is easily remedied by replacing $\mathcal{P}$ with a bounded subfunctor $\mathcal{P}_\alpha X = \{Y \subseteq X : |Y| < \alpha\}$ for some regular cardinal α.

The main feature of abstract GSOS is its compositionality result: behavioural equivalence of processes with respect to the final coalgebra νB is preserved by all program contexts. To make this statement formal, we use the following categorical concept:

▶ **Definition 4.3** (Morphic Congruence)**.** A *morphic congruence* on a Σ-algebra (A, a) is a morphism $h\colon A \to A'$ of $\mathcal{C}$ that carries a Σ-algebra morphism, that is, there exists a Σ-algebra structure (A', a') on A' such that $h\colon (A, a) \to (A', a')$ is a morphism of Σ-algebras.

▶ Remark 4.4. For a polynomial set functor Σ, if $h\colon A \to A'$ is a morphic congruence then its kernel $\equiv \subseteq A \times A$ given by $a \equiv a'$ iff $h(a) = h(a')$ forms a congruence relation on A in the usual sense: for all n-ary f in Σ, if $a_i \equiv a'_i$ for $i = 1, \dots, n$ then $\mathsf{f}^A(a_1, \dots, a_n) \equiv \mathsf{f}^A(a'_1, \dots, a'_n)$.

▶ **Theorem 4.5** (Congruence Theorem for Pre-GSOS Laws)**.** *Let ϱ be a pre-GSOS law that is natural w.r.t. the morphisms $\hat{\iota}\colon \Sigma^*(\mu\Sigma) \to \mu\Sigma$ and $\Sigma^*\mathsf{beh}\colon \Sigma^*(\mu\Sigma) \to \Sigma^*(\nu B)$. Then the behaviour morphism $\mathsf{beh}\colon \mu\Sigma \to \nu B$ is a morphic congruence on the initial algebra $(\mu\Sigma, \iota)$.*

▶ **Example 4.6.** When applied to the setting of Example 4.2, the above theorem implies the congruence result for the GSOS format by Bloom et al. [11]: for every set of GSOS rules, behavioural equivalence on its operational model (4.4) (i.e. the equivalence relation $\equiv \subseteq \mu\Sigma \times \mu\Sigma$ given by $t \equiv s$ iff $\mathsf{beh}(t) = \mathsf{beh}(s)$) is a congruence w.r.t. all constructors from Σ. Note that behavioural equivalence coincides with *strong bisimilarity* for LTS [40].

Theorem 4.5 is a refined version of Turi and Plotkin's congruence result for GSOS laws [44]. Their original proof involves liftings of comonads to categories of algebras. It is conceptually elegant, but inherently reliant on naturality and thus not applicable to possibly non-natural pre-GSOS laws. Our proof of Theorem 4.5, therefore takes a more direct route that allows to analyse the required naturality conditions and make them explicit. The remainder of this section is devoted to sketching the proof.

▶ **Notation 4.7.** Every pre-GSOS law (4.1) induces a family of morphisms

$$\varrho^*_X\colon \Sigma^*(X \times BX) \to B\Sigma^* X \qquad (X \in \mathcal{C}) \tag{4.5}$$

defined recursively (using the universal property of the free algebra $\Sigma^*(X \times BX)$) as the unique morphism making the diagram below commute:

$$\begin{array}{ccccccc}
X \times BX & \xrightarrow{\eta} & \Sigma^*(X \times BX) & \xleftarrow{\quad\iota\quad} & \Sigma\Sigma^*(X \times BX) \\
{\scriptstyle \mathsf{outr}}\downarrow & & \downarrow{\scriptstyle \varrho^*} & & \downarrow{\scriptstyle \Sigma\langle \Sigma^*\mathsf{outl},\, \varrho^*\rangle} \\
BX & \xrightarrow{B\eta_X} & B\Sigma^* X & \xleftarrow{B\mu} & B\Sigma^*\Sigma^* X \xleftarrow{\varrho} \Sigma(\Sigma^* X \times B\Sigma^* X)
\end{array}$$

Informally, while the original law ϱ specifies the behaviour of individual operations from Σ (corresponding to flat terms), the extension ϱ^* extends this specification to arbitrary terms. Importantly, this extension interacts well with naturality conditions on ϱ. This is the content of the next two lemmas:

▶ **Lemma 4.8.** *For every $f\colon X \to Y$, if the pre-GSOS law ϱ (4.1) is natural w.r.t. $\Sigma^* f$, then its extension ϱ^* (4.5) is natural w.r.t. f.*

▶ **Lemma 4.9.** *If a pre-GSOS law ϱ is natural w.r.t. $\hat{\iota}$, then the following diagram commutes, where γ is the operational model of ϱ:*

$$\begin{array}{ccccc}
\mu\Sigma & \xleftarrow{\quad\hat{\iota}\quad} & \Sigma^*(\mu\Sigma) & & \\
{\scriptstyle \gamma}\downarrow & & \downarrow{\scriptstyle \Sigma^*\langle \mathsf{id},\, \gamma\rangle} & & \\
B(\mu\Sigma) & \xleftarrow{B\hat{\iota}} & B\Sigma^*(\mu\Sigma) & \xleftarrow{\varrho^*} & \Sigma^*(\mu\Sigma \times B(\mu\Sigma))
\end{array}$$

Proof of Theorem 4.5 (Sketch). Every B-coalgebra (X, c) induces a B-coalgebra structure $\overline{c}$ on $\Sigma^* X$ defined as follows:

$$\overline{c} = (\, \Sigma^* X \xrightarrow{\Sigma^* \langle \mathsf{id}, c \rangle} \Sigma^*(X \times BX) \xrightarrow{\varrho_X^*} B\Sigma^* X \,).$$

In particular, for the final coalgebra $(\nu B, \tau)$ we obtain the B-coalgebra $(\Sigma^*(\nu B), \overline{\tau})$. Thus, we get a Σ^*-algebra structure on νB (equivalent to a Σ-algebra structure) given by unique coalgebra morphism

$$\alpha \colon (\Sigma^*(\nu B), \overline{\tau}) \to (\nu B, \tau).$$

To prove that beh is a morphic congruence, it suffices to show that

$$\mathsf{beh} \colon (\mu\Sigma, \iota) \to (\nu B, \alpha)$$

is a Σ^*-algebra morphism, that is, $\mathsf{beh} \circ \hat{\iota} = \alpha \circ \Sigma^* \mathsf{beh}$. The latter equality follows by finality of the coalgebra νB once we show that all four morphisms in the square below are B-coalgebra morphisms, which can be checked using Lemmas 4.8 and 4.9. ◀

$$\begin{array}{ccc} (\Sigma^*(\mu\Sigma), \overline{\gamma}) & \xrightarrow{\hat{\iota}} & (\mu\Sigma, \gamma) \\ {\scriptstyle \Sigma^* \mathsf{beh}}\downarrow & & \downarrow{\scriptstyle \mathsf{beh}} \\ (\Sigma^*(\nu B), \overline{\tau}) & \xrightarrow{\alpha} & (\nu B, \tau) \end{array}$$

5 De Simone Laws and Compositional Coalgebraic Trace Semantics

Our aim is to use Theorem 4.5 to reason about congruence properties of coalgebraic trace semantics (Section 3). For this purpose, we introduce *De Simone laws*, a subclass of GSOS laws designed to smoothly extend to Kleisli categories. Here the behaviour map corresponds to the trace map, making the theory of the previous section applicable to trace semantics.

5.1 De Simone Rules

To motivate our approach, we consider *De Simone rules* [17], a type of GSOS rules that is known to guarantee congruence of trace semantics [10] rather than just strong bisimilarity.

▶ **Definition 5.1** (De Simone Rule)**.** Fix a signature Σ, a set L of labels, and infinitely many variables x_i and y_i $(i \in \mathbb{N})$. A *De Simone rule* is a GSOS rule of the form

$$\frac{x_{i_1} \xrightarrow{a_1} y_{i_1} \quad \cdots \quad x_{i_k} \xrightarrow{a_k} y_{i_k}}{\mathsf{f}(x_1, \ldots, x_n) \xrightarrow{a} t} \tag{5.1}$$

where (i) $\mathsf{f} \in \Sigma$ is an n-ary operation, (ii) $i_1, \ldots, i_k \in \{1, \ldots, n\}$ are pairwise distinct, (iii) $a_1, \ldots, a_k, a \in L$, and (iv) t is an affine Σ-term in the variables y_i $(i \in \{i_1, \ldots, i_k\})$ and x_j $(j \in \{1, \ldots, n\} \setminus \{i_1, \ldots, i_k\})$. (A term is *affine* if no variable occurs more than once in it.)

▶ **Example 5.2.** The two rules (4.2) of parallel composition are De Simone rules up to renaming the variables p, p', q, q' to x_1, y_1, x_2, y_2. A non-example is the rule (5.2) below, whose output term $\mathsf{f}(y_1, y_1)$ is not affine because the variable y_1 appears twice in it.

$$\frac{x_1 \xrightarrow{a_1} y_1}{\mathsf{f}(x_1, x_2) \xrightarrow{a} \mathsf{f}(y_1, y_1)} \tag{5.2}$$

Since De Simone rules are GSOS rules, they can be presented as GSOS laws of Σ over $BX = \mathcal{P}(L\times X)$, viz. natural transformations $\varrho_X\colon \Sigma(X\times\mathcal{P}(L\times X)) \to \mathcal{P}(L\times\Sigma^*X)$. However, the specific form of De Simone rules allows for a simpler kind of natural transformation:

▶ **Construction 5.3.** Every set $\mathcal{R}$ of De Simone rules induces a natural transformation

$$\varrho_X\colon \Sigma(X + L\times X + 1) \to \mathcal{P}(L\times\Sigma^*X + 1) \qquad (X \in \mathbf{Set}) \tag{5.3}$$

where ϱ_X maps $\mathsf{f}(u_1,\dots,u_n) \in \Sigma(X + L\times X + 1)$ to the following subset of $L\times\Sigma^*X + 1$:

- If $u_i = *$ for some i, then $\varrho_X(\mathsf{f}(u_1,\dots,u_n)) = \{*\}$.
- Otherwise let $\{i_1\dots,i_k\} = \{i \mid u_i \in L\times X\}$; thus $u_{i_l} = (a_l, v_{i_l})$ for some $a_l \in L$ and $v_{i_l} \in X$, and $u_j \in X$ for $j \notin \{i_1,\dots,i_k\}$. Then $\varrho_X(\mathsf{f}(u_1,\dots,u_n))$ consists of $*$ and all pairs $(a, t[\sigma]) \in L\times\Sigma^*X$ such that $\mathcal{R}$ contains the rule (5.1) and $t[\sigma]$ emerges from the term t via the substitution σ given by $y_{i_l} \mapsto v_{i_l}$ and $x_j \mapsto u_j$ for $j \notin \{i_1,\dots,i_k\}$.

Note that the "+1" component flags all states as terminating, as always $* \in \varrho_X(\mathsf{f}(u_1,\dots,u_n))$. This is necessary to fit the coalgebraic trace semantics framework (cf. Example 3.1), and makes sure that all emerging partial traces are being tracked.

5.2 De Simone Laws

The above modelling of De Simone rules as natural transformations of type (5.3) immediately suggests a categorical generalization. In the remainder, let us fix the following data:

- a category $\mathcal{C}$ with finite limits and finite coproducts;
- two endofunctors Σ, B and a monad T on $\mathcal{C}$, where Σ has an initial algebra $(\mu\Sigma, \iota)$ and free algebras Σ^*X.

▶ **Definition 5.4** (Pre-De Simone Law)**.** A *pre-De Simone law* is a natural transformation

$$\varrho_X\colon \Sigma(X + BX) \to TB\Sigma^*X \qquad (X \in \mathcal{C}). \tag{5.4}$$

De Simone rules thus correspond to pre-De Simone laws with $\mathcal{C} = \mathbf{Set}$, $BX = L\times X + 1$, $T = \mathcal{P}$, and a polynomial functor Σ. To reason about abstract congruence properties for trace semantics of pre-De Simone laws, the given data needs some additional structure:

▶ **Assumptions 5.5.** In the following, we assume that:

(1) There is a natural isomorphism $j\colon TX\times TY \xrightarrow{\cong} T(X+Y)$ $(X, Y \in \mathcal{C})$.

(2) There are functor-over-monad distributive laws of Σ and B over T. We denote the laws and their Kleisli extensions by $\delta^\Sigma\colon \Sigma T \to T\Sigma$, $\delta^B\colon BT \to TB$, and $\overline{\Sigma}, \overline{B}\colon \mathbf{Kl}(T) \to \mathbf{Kl}(T)$.

(3) T, B, $\overline{B}$ form a setting for coalgebraic trace semantics (Definition 3.2).

▶ Remark 5.6. Condition (1) entails that $\mathbf{Kl}(T)$ has binary products, which coincide with binary coproducts in $\mathbf{Kl}(T)$ and thus binary coproducts in $\mathcal{C}$ [16, Thm. 19]. Monads T satisfying (1) and additionally $T0 \cong 1$ are known as *additive monads* [16].

▶ Remark 5.7. The distributive law δ^B extends to a functor-over-monad distributive law of $B_0X = X + BX$ over T and thus a corresponding Kleisli extension of B_0:

$$\delta^{B_0} = (\, B_0TX \xrightarrow{\mathsf{id}+\delta^B} TX + TBX \xrightarrow{[T\mathsf{inl}, T\mathsf{inr}]} TB_0X \,) \qquad \text{and} \qquad \overline{B}_0\colon \mathbf{Kl}(T) \to \mathbf{Kl}(T).$$

▶ Remark 5.8. By Proposition 2.3, the free monad $\overline{\Sigma^*}$ of $\overline{\Sigma}$ is a Kleisli extension of the free monad Σ^* of Σ. We write $\delta^{\Sigma^*}\colon \Sigma^*T \to T\Sigma^*$ for the ensuing distributive law extending δ^Σ.

Since products coincide with coproducts in $\mathbf{Kl}(T)$ (Remark 5.6), every pre-De Simone law (5.4) forms a pre-GSOS law of $\overline{\Sigma}$ over $\overline{B}$ in $\mathbf{Kl}(T)$, that is, a family of morphisms

$$\varrho_X \colon \overline{\Sigma}(X \times \overline{B}X) \multimap \overline{B}\overline{\Sigma^*}X \qquad (X \in \mathbf{Kl}(T)). \tag{5.5}$$

Additionally, a pre-De Simone law yields a GSOS law in $\mathcal{C}$ with the same operational model:

▶ **Proposition 5.9.** *Every pre-De Simone law* (5.4) *induces a GSOS law of* Σ *over* TB *in* $\mathcal{C}$*:*

$$\begin{array}{ccccccc} \Sigma(X \times TBX) & & & & & \xrightarrow{\bar{\varrho}_X} & TB\Sigma^*X \\ {\scriptstyle \Sigma(\eta_X \times \mathsf{id})}\downarrow & & & & & & \uparrow{\scriptstyle \mu_{B\Sigma^*X}} \\ \Sigma(TX \times TBX) & \xrightarrow{\Sigma j} & \Sigma T(X + BX) & \xrightarrow{\delta^{\Sigma}_{X+BX}} & T\Sigma(X + BX) & \xrightarrow{T\varrho_X} & TTB\Sigma^*X \end{array} \tag{5.6}$$

The laws ϱ (5.5) *and* $\bar{\varrho}$ (5.6) *have the same operational model* $\gamma \colon \mu\Sigma \to TB(\mu\Sigma)$*.*

Note that (5.5) is generally not a GSOS law: naturality of (5.4) in $\mathcal{C}$ corresponds to naturality of (5.5) w.r.t. *pure* morphisms, but not arbitrary morphisms in $\mathbf{Kl}(T)$. To apply the general congruence result for pre-GSOS laws (Theorem 4.5) to the pre-De Simone law (5.5), we need additional naturality conditions beyond pure morphisms. We use the following terminology:

▶ **Definition 5.10.** A pre-De Simone law (5.4) is *natural* w.r.t. a morphism $p \colon P \multimap X$ in $\mathbf{Kl}(T)$ if the corresponding pre-GSOS law (5.5) in $\mathbf{Kl}(T)$ is natural w.r.t. p.

The following proposition gives an elementary characterization of naturality. It is this condition that we will verify in our applications in Sections 6 and 7.

▶ **Proposition 5.11.** *A pre-De Simone law* (5.4) *is natural w.r.t.* $p \colon P \multimap X$ *iff the rectangle in* $\mathcal{C}$ *shown below commutes when precomposed with* $\Sigma B_0 p$ *and postcomposed with* μ*:*

$$\begin{array}{ccccccc} \Sigma B_0 P & \xrightarrow{\Sigma B_0 p} & \Sigma B_0 TX & \xrightarrow{\Sigma \delta^{B_0}} & \Sigma T B_0 X & \xrightarrow{\delta^{\Sigma}} & T\Sigma B_0 X \\ & & {\scriptstyle \varrho_{TX}}\downarrow & & & & \downarrow{\scriptstyle T\varrho_X} \\ & & TB\Sigma^*TX & \xrightarrow{TB\delta^{\Sigma^*}} & TBT\Sigma^*X & \xrightarrow{T\delta^B} & TTB\Sigma^*X \xrightarrow{\mu} TB\Sigma^*X \end{array} \tag{5.7}$$

5.3 Compositionality

For every pre-De Simone law (5.4) with its corresponding pre-GSOS law (5.5) in $\mathbf{Kl}(T)$, let us denote the operational model and its associated trace (i.e. behaviour) morphism by $\gamma \colon \mu\overline{\Sigma} \multimap \overline{B}(\mu\overline{\Sigma})$ and $\mathsf{tr} \colon \mu\overline{\Sigma} \multimap \nu\overline{B} = \mu\overline{B}$. Our congruence result for this setting (Theorem 5.15 below) involves another monad T_+ alongside T, thought of as the "positive" part of T. For example, for $T = \mathcal{P}$ one takes the submonad $T_+ = \mathcal{P}_+$, the non-empty power set monad. This is an instance of a general construction for computing the *affine part* of a monad[1]. Recall that we assume $\mathcal{C}$ to have finite limits.

▶ **Proposition 5.12** ([36])**.** *For every* $X \in \mathcal{C}$*, let* $i_X \colon T_+X \to TX$ *be the equalizer*

$$T_+X \xrightarrow{i_X} TX \underset{T!}{\overset{\eta \circ !}{\rightrightarrows}} T1.$$

Then T_+ *extends to an affine monad and the family* i_X $(X \in \mathcal{C})$ *to a monad morphism.*

[1] A monad T is *affine* if $T1 \cong 1$. We warn the reader that this notion is not connected to the affineness of terms in the definition of De Simone rules.

Let us call a morphism $p\colon P \to TX$ *affine* if it factors through $i_X\colon T_+X \to TX$. From the universal property of the equalizer i_X, the following is immediate:

▶ **Lemma 5.13.** *A morphism $p\colon P \to TX$ is affine iff $T! \circ p = \eta \circ !$.*

▶ **Definition 5.14** (De Simone Law)**.** A *De Simone law* is a pre-De Simone law (5.4) that is natural w.r.t. the $\mathbf{Kl}(T)$-morphism $i_X\colon T_+X \nrightarrow X$ for each $X \in \mathcal{C}$ (cf. Definition 5.10).

This leads to our main result, a general compositionality criterion for De Simone laws:

▶ **Theorem 5.15** (Compositionality of Coalgebraic Trace Semantics)**.** *For every De Simone law* (5.4) *whose trace morphism* $\mathsf{tr}\colon \mu\Sigma \to T(\mu B)$ *is affine,* tr *is a morphic Σ-congruence.*

The proof (given in the full arXiv paper) uses the congruence result for pre-GSOS laws (Theorem 4.5) and amounts to verifying that the naturality conditions of the latter follow from the naturality of the De Simone law (5.4) w.r.t. $i_X\colon T_+X \nrightarrow X$ and the affineness of tr.

We shall see below that the naturality properties of (5.4) reflect precisely the requirement of affineness of output terms in De Simone rules (Definition 5.1). In contrast, affineness of the trace morphism tr is a non-trivial global property. For example, in a setting of probabilistic specifications (Section 7) it expresses that the operational model is *almost surely terminating*. On a technical level, by Lemma 5.13, affineness of tr amounts to the equality $T! \circ \mathsf{tr} = \eta \circ !$. In some instances, one of the inequalities constituting it follows nearly for free:

▶ **Proposition 5.16.** *If the morphism $1 \xrightarrow{\langle \eta,\eta\rangle} T1\times T1 \xrightarrow{j} T(1+1)$ and every component* (5.4) *is affine, then so is γ, and moreover the induced trace morphism* tr *satisfies $T! \circ \mathsf{tr} \sqsubseteq \eta \circ !$.*

6 Case Study I: De Simone Rules

As a first showcase of our abstract theory, we derive the congruence of De Simone rules.

6.1 Categorical Setting

Recall that De Simone rules yield pre-De Simone laws for $\mathcal{C} = \mathbf{Set}$, $BX = L \times X + 1$, $T = \mathcal{P}$, and a polynomial functor Σ. Let us verify that this data satisfies the Assumptions 5.5:

(1) We have the natural isomorphism $j\colon \mathcal{P}X \times \mathcal{P}Y \cong \mathcal{P}(X + Y)$ given by $j(S,T) = S \uplus T$.

(2) The distributive law $\delta^B\colon L \times \mathcal{P}X + 1 \to \mathcal{P}(L \times X + 1)$ is given by Example 3.4, and the law $\delta^\Sigma\colon \Sigma\mathcal{P} \to \mathcal{P}\Sigma$ by $\mathsf{f}(X_1,\dots,X_n) \mapsto \{\,\mathsf{f}(x_1,\dots,x_n) \mid x_i \in X_i \text{ for } i = 1,\dots,n\,\}$.

(3) $\mathcal{P}, B, \bar{B}$ form a setting for coalgebraic trace semantics, as explained in Example 3.4.

The affine part of $\mathcal{P}$ is the non-empty power set monad $\mathcal{P}_+$. Thus a morphism $p\colon P \nrightarrow X$ of $\mathbf{Kl}(\mathcal{P})$ is affine iff $p(z) \neq \emptyset$ for all $z \in P$, that is, p corresponds to a right-total relation.

6.2 Compositionality

The key lies in the next proposition, whose proof illustrates how the affineness of terms in De Simone rules corresponds to the induced pre-De Simone law being a (proper) De Simone law:

▶ **Proposition 6.1.** *For every set of De Simone rules,* (5.3) *is a De Simone law.*

Proof sketch. By Proposition 5.11, we are to show that the two legs of the diagram below send any element $\mathsf{f}(U_1,\dots,U_n)$ of $\Sigma(\mathcal{P}_+X + L \times \mathcal{P}_+X + 1)$ to the same element of $\mathcal{P}(L \times \Sigma^*X + 1)$.

$$\begin{array}{ccccccc}
\Sigma(\mathcal{P}X + L \times \mathcal{P}X + 1) & \xrightarrow{\Sigma\delta^{B_0}} & \Sigma\mathcal{P}(X + L \times X + 1) & \xrightarrow{\delta^\Sigma} & \mathcal{P}\Sigma(X + L \times X + 1) & & \\
\downarrow{\scriptstyle \varrho_{\mathcal{P}X}} & & & & \downarrow{\scriptstyle \mathcal{P}\varrho_X} & & \\
\mathcal{P}(L \times \Sigma^*\mathcal{P}X + 1) & \xrightarrow{\mathcal{P}B\delta^{\Sigma^*}} & \mathcal{P}(L \times \mathcal{P}\Sigma^*X + 1) & \xrightarrow{\mathcal{P}\delta^B} & \mathcal{P}\mathcal{P}(L \times \Sigma^*X + 1) & \xrightarrow{\mu} & \mathcal{P}(L \times \Sigma^*X + 1)
\end{array}$$

To illustrate the gist of the argument, let us consider a binary operator f and assume that the only rule for f is given by (6.1) below, for fixed labels $a, a_1 \in L$:

$$\frac{x_1 \xrightarrow{a_1} y_1}{\mathsf{f}(x_1,x_2) \xrightarrow{a} \mathsf{f}(y_1,x_2)} \tag{6.1}$$

The interesting case is that of an element $\mathsf{f}(U_1,U_2)$ with $U_1 = (a_1,V_1) \in L \times \mathcal{P}_+X$ and $U_2 \in \mathcal{P}_+X$. The map $\varrho_{\mathcal{P}X}$ sends $\mathsf{f}(U_1,U_2)$ to the set $\{*,(a,\mathsf{f}(V_1,U_2))\} \in \mathcal{P}(L \times \Sigma^*\mathcal{P}X + 1)$. The pair $(a,\mathsf{f}(V_1,U_2))$ is mapped by $\delta^B \circ B\delta^{\Sigma^*}$ to the set of all pairs $(a,\mathsf{f}(v_1,u_2))$ where $v_1 \in V_1$ and $u_2 \in U_2$. (Here affineness of the term $\mathsf{f}(y_1,x_2)$ in (6.1) is used, see Remark 6.2.) Thus, overall, $\mu \circ \mathcal{P}\delta^B \circ \mathcal{P}B\delta^{\Sigma^*} \circ \varrho_{\mathcal{P}X}(\mathsf{f}(U_1,U_2))$ is the subset of $L \times \Sigma^*X + 1$ consisting of:

(1) the element $*$ of 1; (2) all pairs $(a,\mathsf{f}(v_1,u_2)) \in L \times \Sigma^*X$ with $v_1 \in V_1$ and $u_2 \in U_2$.

Similarly, $\mathsf{f}(U_1,U_2)$ is mapped by $\delta^\Sigma \circ \Sigma\delta^{B_0}$ to the non-empty set of all $\mathsf{f}((a_1,v_1),u_2)$ with $v_1 \in V_1$ and $u_2 \in U_2$. Moreover, ϱ_X maps each such $\mathsf{f}((a_1,v_1),u_2)$ to the set $\{*,(a,\mathsf{f}(v_1,u_2))\}$. Thus $\mu \circ \mathcal{P}\varrho_X \circ \delta^\Sigma \circ \Sigma\delta^{B_0}(\mathsf{f}(U_1,U_2))$ is also given by the above elements (1) and (2). ◀

▶ Remark 6.2. Some subtleties of the above proof are worth mentioning:

(1) The reasoning critically rests on affineness of the output term $\mathsf{f}(y_1,x_2)$ of the rule (6.1) and fails for non-affine terms. For example, suppose that f is instead specified by the rule (5.2) with the non-affine output term $\mathsf{f}(y_1,y_1)$. Now
- $\mu \circ \mathcal{P}\delta^B \circ \mathcal{P}B\delta^{\Sigma^*} \circ \varrho_{\mathcal{P}X}(\mathsf{f}(U_1,U_2))$ consists of $*$ and all $\mathsf{f}(v_1,v_1')$ with $v_1,v_1' \in V_1$.
- $\mu \circ \mathcal{P}\varrho \circ \delta^\Sigma \circ \Sigma\delta^{B_0}(\mathsf{f}(U_1,U_2))$ consists of $*$ and all $\mathsf{f}(v_1,v_1)$ with $v_1 \in V_1$.

These two sets differ whenever V_1 has more than one element.

(2) It is also critical that the sets V_1 and U_2 considered in the proof are *non-empty*. For example, suppose that $V_1 = \emptyset$ (so $U_1 = (a_1,\emptyset)$). Then

$$\mu \circ \mathcal{P}\delta^B \circ \mathcal{P}B\delta^{\Sigma^*} \circ \varrho(\mathsf{f}(U_1,U_2)) = \{*\} \qquad \text{while} \qquad \mu \circ \mathcal{P}\varrho \circ \delta^\Sigma \circ \Sigma\delta^{B_0}(\mathsf{f}(U_1,U_2)) = \emptyset.$$

Recall that we need "$*$" to signal potential trace termination (Construction 5.3).

From the above proposition the congruence result for De Simone rules is now immediate. Let $\mathcal{R}$ be a set of such rules with operational model $\gamma\colon \mu\Sigma \to \mathcal{P}(L \times \mu\Sigma + 1)$ and trace map $\mathsf{tr}\colon \mu\Sigma \to \mathcal{P}(\mu B) = \mathcal{P}(L^*)$. Two terms $t,s \in \mu\Sigma$ are *trace equivalent* if $\mathsf{tr}(t) = \mathsf{tr}(s)$.

▶ **Theorem 6.3** (Compositionality of De Simone Rules [10]). *For every set of De Simone rules, trace equivalence on the operational model is a congruence relation.*

Proof. The trace map tr is affine since $\varepsilon \in \mathsf{tr}(t)$ for all t, so by Proposition 6.1 and Theorem 5.15 it forms a morphic congruence. By Remark 4.4, this implies that trace equivalence (the kernel of tr) forms a congruence relation. ◀

7 Case Study II: Probabilistic De Simone Rules

Thanks to the categorical generality of De Simone laws, varying the underlying monad T leads to novel De Simone-type congruence formats for LTS with computational effects that are more complex than non-determinism. For illustration, we move to a probabilistic setting.

7.1 Weighted and Probabilistic Labelled Transition Systems

The systems under consideration are a weighted form of LTS, which involve a weighted generalization of the power set monad. Consider the endofunctor $\mathcal{W}\colon \mathbf{Set} \to \mathbf{Set}$ given by

$$\mathcal{W}X = [0,\infty]^X, \qquad \mathcal{W}(f\colon X \to Y)\colon \sum_{i\in I} r_i \cdot x_i \mapsto \sum_{i\in I} r_i \cdot f(x_i).$$

Here a function $\phi \in \mathcal{W}X$ is identified with a formal sum $\sum_{i\in I} r_i \cdot x_i$ where $r_i \in [0,\infty]$, $x_i \in X$, and $\sum_{x_i = x} r_i = \phi(x)$ for all $x \in X$. The functor $\mathcal{W}$ extends to a monad with unit $\eta_X\colon X \to \mathcal{W}X$ and multiplication $\mu_X\colon \mathcal{W}\mathcal{W}X \to \mathcal{W}X$ defined as follows:

$$\eta_X\colon x \mapsto 1 \cdot x, \qquad \mu_X\colon \sum_{i\in I} r_i \cdot \Big(\sum_{j\in J_i} r_{ij} \cdot x_{ij}\Big) \mapsto \sum_{i\in I}\sum_{j\in J_i} r_i \cdot r_{ij} \cdot x_{ij}.$$

The affine part $\mathcal{W}_+$ is the *distribution monad* $\mathcal{D}X = \{\phi \in \mathcal{W}X \mid \sum_{x\in X} \phi(x) = 1\}$ formed by discrete probability distributions. Thus a morphism $p\colon P \mathrel{-\!\!\circ\!\!\to} X$ in $\mathbf{Kl}(\mathcal{W})$ is affine iff $p(z) \in \mathcal{D}X$ for all $z \in P$. A *$\mathcal{W}$-LTS* (with explicit termination) is a coalgebra

$$c\colon C \to \mathcal{W}(L \times C + 1). \tag{7.1}$$

We write $x \xrightarrow{a/r} y$ if $c(x)(a,y) = r$, and $x \xrightarrow{r} *$ if $c(x)(*) = r$. A *(generative) probabilistic LTS* [25] is a $\mathcal{W}$-LTS whose structure c is affine ($c(x) \in \mathcal{D}(L \times C + 1)$ for $x \in C$). Here, weights are interpreted as probabilities: $x \xrightarrow{a/r} y$ means that with probability r the process x outputs a and transitions into y, and $x \xrightarrow{r} *$ means that x terminates with probability r.

▶ Remark 7.1. We stick to $[0,\infty]$ for simplicity. All present definitions and results extend to weights from any continuous commutative semiring [18]. (Continuity is needed to satisfy the assumptions of coalgebraic trace semantics [27], specifically $\mathbf{DCPO}_\perp$-enrichment of the Kleisli category.) In particular, taking the boolean semiring $\{0,1\}$ (with $\vee$ as addition and $\wedge$ as multiplication) recovers the results of Section 6.

The *trace map* of a $\mathcal{W}$-LTS (7.1) is the map $\mathsf{tr}\colon C \to \mathcal{W}(L^*)$ given by

$$\mathsf{tr}(x)(a_1 \cdots a_n) = \sum_{x = x_0,\dots,x_n} \Big(\prod_{i=0}^{n-1} c(x_i)(a_{i+1}, x_{i+1})\Big) \cdot c(x_n)(*), \tag{7.2}$$

with the sum ranging over all paths of length n starting from x, that is, all lists $x_0,\dots,x_n$ of elements of C where $x = x_0$. In the probabilistic case, $\mathsf{tr}(x)(a_1 \cdots a_n)$ is the probability that the process x generates the trace $a_1 \cdots a_n$ and then terminates. This definition is captured by the setting T, B, $\overline{B}$ for coalgebraic trace semantics where $T = \mathcal{W}$ and $BX = L \times X + 1$, and the Kleisli extension $\overline{B}\colon \mathbf{Kl}(\mathcal{W}) \to \mathbf{Kl}(\mathcal{W})$ corresponds to the distributive law

$$\delta^B\colon L \times \mathcal{W}X + 1 \to \mathcal{W}(L \times X + 1), \qquad (a, \sum_{i\in I} r_i \cdot x_i) \mapsto \sum_{i\in I} r_i \cdot (a, x_i), \qquad * \mapsto 1 \cdot *.$$

Thus $\overline{B}$ sends $f\colon X \mathrel{-\!\!\circ\!\!\to} Y$ to $\overline{B}f\colon L \times X + 1 \mathrel{-\!\!\circ\!\!\to} L \times Y + 1$ given by

$$(a,x) \mapsto \sum_{i\in I} r_i \cdot (a, y_i) \qquad \text{and} \qquad * \mapsto 1 \cdot *, \tag{7.3}$$

where $f(x) = \sum_{i\in I} r_i \cdot y_i$. Then all conditions of Definition 3.2 are satisfied:

(1) B has the initial algebra $(\mu B, \beta)$ carried by L^*, as before.

(2) The category $\mathbf{Kl}(\mathcal{W})$ is left strictly $\mathbf{DCPO}_\perp$-enriched: for $f, g\colon X \mathrel{-\!\!\circ\!\!\to} Y$ the order is given by $f \sqsubseteq g$ iff $f(x) \le g(x)$ for all $x \in X$, where $\le$ is the usual order of the interval $[0,\infty]$.

(3) The lifting $\overline{B}$ is locally monotone; this is clear from (7.3).

For every $\mathcal{W}$-LTS (7.1), the coalgebraic trace map $\mathsf{tr}\colon C \mathrel{-\!\!\circ\!\!\to} \nu\overline{B} = \mu\overline{B} = L^*$ is given by (7.2). Indeed, one readily verifies that (7.2) is a coalgebra morphism from (C, c) to $(\mu B, J\beta^{-1})$.

7.2 A Probabilistic Rule Format

We now introduce a probabilistic version of De Simone rules. Its syntax and semantics are closely related to *probabilistic GSOS* [8] and *stochastic GSOS* [31, 32]. The main difference is that the latter apply to reactive probabilistic systems, while our rules apply to generative ones, which is the more natural setting for trace semantics. In addition, we put restrictions on the premises and the output terms of rules similar to the original De Simone format.

▶ **Definition 7.2** (Weighted De Simone Rule)**.** Fix a signature Σ, label set L, and variables x_i, y_i as in Definition 5.1. A *weighted De Simone rule* is a rule of either of the two forms

$$\frac{x_{i_1} \xrightarrow{a_1} y_{i_1} \quad \cdots \quad x_{i_k} \xrightarrow{a_k} y_{i_k} \quad x_{j_1} \to * \quad \cdots \quad x_{j_m} \to *}{\mathsf{f}(x_1,\dots,x_n) \xrightarrow{a/r} t} \tag{7.4}$$

$$\frac{x_{i_1} \xrightarrow{a_1} y_{i_1} \quad \cdots \quad x_{i_k} \xrightarrow{a_k} y_{i_k} \quad x_{j_1} \to * \quad \cdots \quad x_{j_m} \to *}{\mathsf{f}(x_1,\dots,x_n) \xrightarrow{r} *} \tag{7.5}$$

where (i) f is an n-ary operation symbol from Σ, (ii) the indices $i_1,\dots,i_k,j_1,\dots,j_m \in \{1,\dots,n\}$ are pairwise distinct, (iii) $a_1,\dots,a_k,a \in L$, (iv) $r \in [0,\infty]$, and (v) t is an affine Σ-term in the variables y_i $(i \in \{i_1,\dots,i_k\})$ and x_j $(j \in \{1,\dots,n\} \setminus \{i_1,\dots,i_k,j_1,\dots,j_m\})$.

▶ Remark 7.3. Weighted De Simone rules (7.4) could be equivalently but semantically more suggestively rewritten as

$$\frac{x_{i_1} \xrightarrow{a_1/r_1} y_{i_1} \quad \cdots \quad x_{i_k} \xrightarrow{a_k/r_k} y_{i_k} \quad x_{j_1} \xrightarrow{r'_1} * \quad \cdots \quad x_{j_m} \xrightarrow{r'_m} *}{\mathsf{f}(x_1,\dots,x_n) \xrightarrow{a/r\cdot r_1\cdot\dots\cdot r_n\cdot r'_1\cdot\dots\cdot r'_m} t}$$

with variables r_i, r'_j representing weights; similarly for (7.5). This indicates how the transition weights of the premises contribute to the transition weight in the conclusion. More formally:

▶ **Definition 7.4** (Operational Model)**.** Given a set $\mathcal{R}$ of weighted De Simone rules, the *operational model* of $\mathcal{R}$ is the coalgebra

$$\gamma\colon \mu\Sigma \to \mathcal{W}(L \times \mu\Sigma + 1) \tag{7.6}$$

defined by structural recursion as follows. Given a term $\mathsf{f}(t_1,\dots,t_n)$ in $\mu\Sigma$ and a rule $R \in \mathcal{R}$ for f, we let $\gamma_R(\mathsf{f}(t_1,\dots,t_n)) \in \mathcal{W}(L \times \mu\Sigma + 1)$ denote the weighted sum

$$\begin{cases} \sum_{s_{i_1},\dots,s_{i_k} \in \mu\Sigma} r \cdot \prod_{l=1}^{k} \gamma(t_{i_l})(a_l, s_{i_k}) \cdot \prod_{l=1}^{m} c(t_{j_l})(*) \cdot (a, t[\sigma]) & \text{if } R \text{ is given by (7.4);} \\ \sum_{s_{i_1},\dots,s_{i_k} \in \mu\Sigma} r \cdot \prod_{l=1}^{k} \gamma(t_{i_l})(a_l, s_{i_k}) \cdot \prod_{l=1}^{m} c(t_{j_l})(*) \cdot * & \text{if } R \text{ is given by (7.5).} \end{cases}$$

Here σ denotes the substitution $x_i \mapsto t_i$ and $y_{i_l} \mapsto s_{i_l}$. Then the coalgebra (7.6) is given by $\gamma(\mathsf{f}(t_1,\dots,t_n)) = \sum_{R \in \mathcal{R} \text{ rule for } \mathsf{f}} \gamma_R(\mathsf{f}(t_1,\dots,t_n))$.

▶ **Definition 7.5** (Probabilistic De Simone Specification)**.** A *probabilistic De Simone specification* is a set of weighted De Simone rules whose model (7.6) is a probabilistic LTS. The specification is *almost surely terminating* (*a.s.t.*) if the trace map tr of (7.6) is affine ($\mathsf{tr}(t) \in \mathcal{D}(L^*)$ for $t \in \mu\Sigma$, i.e. there is no positive probability of generating an infinite trace).

▶ **Example 7.6.** Consider a probabilistic process algebra with a terminating process nil, a prefixing operator $a.(-)$ for each $a \in L$, and a parallel composition operator $\|_r$ for each $r \in [0,1]$. The process $p \|_r q$ progresses one of the processes p or q with probability r or $1-r$, respectively. This is formalized by the a.s.t. probabilistic De Simone specification below:

$$\frac{}{\mathsf{nil} \xrightarrow{1} *} \qquad \frac{}{a.p \xrightarrow{a/1} p} \qquad \frac{p \xrightarrow{a} p'}{p \|_r q \xrightarrow{a/r} p' \|_r q} \qquad \frac{p \to *}{p \|_r q \xrightarrow{r} *}$$

$$\frac{q \xrightarrow{a} q'}{p \parallel_r q \xrightarrow{a/1-r} p \parallel_r q'} \qquad \frac{q \to *}{p \parallel_r q \xrightarrow{1-r} *}$$

A simple structural induction shows that the operational model (7.6) is indeed an a.s.t. probabilistic LTS (see the full arXiv paper for details).

▶ Remark 7.7.

(1) It would be desirable to identify syntactic criteria for a set of weighted De Simone rules to form a probabilistic specification. Such criteria exist for probabilistic GSOS for reactive systems [8], but appear to be harder to come by in the generative case.

(2) Proposition 5.16 would not help with Example 7.6: the components of the associated De Simone law given in Construction 7.8 below are not affine, even though the operational model (7.6) of the law itself is affine, that is, a probabilistic LTS.

7.3 Categorical Setting

To capture weighted De Simone rules within our categorical framework, we take $\mathcal{C} = \mathbf{Set}$, $BX = L \times X + 1$, $T = \mathcal{W}$, and a polynomial functor Σ. This data satisfies the Assumptions 5.5:

(1) We have the natural isomorphism $j \colon \mathcal{W}X \times \mathcal{W}Y \cong \mathcal{W}(X + Y)$ given by

$$j(\sum\nolimits_{i\in I} r_i \cdot x_i, \sum\nolimits_{j\in J} s_j \cdot y_j) = \sum\nolimits_{i\in I} r_i \cdot x_i + \sum\nolimits_{j\in J} s_j \cdot y_j.$$

(2) The distributive law δ^B is given by Example 3.4, and $\delta^\Sigma \colon \Sigma\mathcal{W}X \to \mathcal{W}\Sigma X$ by

$$\mathsf{f}(\phi_1, \dots, \phi_n) \;\mapsto\; \sum\nolimits_{i_1\in I_1} \cdots \sum\nolimits_{i_n\in I_n} r^1_{i_1} \cdots r^n_{i_n} \cdot \mathsf{f}(x^1_{i_1}, \cdots, x^n_{i_n}) \quad \text{where} \quad \phi_j = \sum\nolimits_{i\in I_j} r^j_i \cdot x^j_i.$$

(3) $\mathcal{W}, B, \bar{B}$ form a setting for coalgebraic trace semantics, as explained in Section 7.1.

▶ **Construction 7.8.** Every set $\mathcal{R}$ of weighted De Simone rules induces a pre-De Simone law of Σ over B, i.e. a natural transformation

$$\varrho_X \colon \Sigma(X + L \times X + 1) \to \mathcal{W}(L \times \Sigma^* X + 1) \qquad (X \in \mathbf{Set}) \tag{7.7}$$

defined as follows. Given $\mathsf{f}(u_1, \dots, u_n) \in \Sigma(X + L \times X + 1)$, let $i_1, \dots, i_k$ be those i with $u_i \in L \times X$, say $u_{i_l} = (a_l, v_{i_l})$, and let $j_1, \dots, j_m$ be those j with $u_j = *$. We say that a rule $R \in \mathcal{R}$ *matches* $\mathsf{f}(u_1, \dots, u_n)$ if it is of the form (7.4) or (7.5), that is, its premises match the above indices and labels. Let $r_R = r$ be the output weight of the rule. Moreover let $a_R = a$ and $t_R = t$ be the output label and output term if the rule is of type (7.4). We then put

$$\varrho_X(\mathsf{f}(u_1, \dots, u_n)) = \sum\nolimits_{R\ (7.4)} r_R \cdot t_R[\sigma] + \sum\nolimits_{R\ (7.5)} r_R \cdot *$$

where the sums range, respectively, over all $R \in \mathcal{R}$ of type (7.4) or (7.5) matching $\mathsf{f}(u_1, \dots, u_n)$, and σ is the substitution $x_i \mapsto u_i$ for $i \notin \{i_1, \dots, i_k, j_1, \dots, j_m\}$ and $y_{i_l} \mapsto v_{i_l}$ for $l = 1, \dots, k$.

The operational model of (7.7) is precisely the operational model (7.6) of the given rules.

7.4 Compositionality

As before in the non-deterministic case (cf. Proposition 6.1), to apply our theory of congruence to probabilistic De Simone specifications we need to verify naturality conditions in $\mathbf{Kl}(\mathcal{W})$, which again boil down to the affineness of output terms required by the rule format:

▶ **Proposition 7.9.** *For any probabilistic De Simone specification,* (7.7) *is a De Simone law.*

Our compositionality result for probabilistic trace semantics now readily emerges. Given a probabilistic De Simone specification, the operational model (7.6) induces the trace map $\mathsf{tr}\colon \mu\Sigma \to \mathcal{W}(L^*)$. *Trace equivalence* is, as usual, the kernel relation of this map; thus two terms are trace equivalent if they generate any given completed trace with the same probability. We then obtain the following probabilistic extension of Theorem 6.3:

▶ **Theorem 7.10** (Compositionality of Probabilistic De Simone Rules)**.** *For any a.s.t. probabilistic De Simone specification, trace equivalence on the operational model is a congruence.*

Proof. By Definition 7.5, the trace map tr is affine. By Proposition 7.9 and Theorem 5.15, tr is a morphic congruence, whence trace equivalence is a congruence relation by Remark 4.4. ◀

A.s.t. probabilistic De Simone specifications thus provide a congruence format for probabilistic trace semantics. To our knowledge, no such format has appeared before in the literature.

▶ **Example 7.11** (Non-a.s.t. Specifications)**.** Note the relevance of the a.s.t. assumption in Theorem 7.10. The easiest way to fail it is by involving a rule like $\dfrac{}{c \xrightarrow{a/1} c}$. The probability of termination for the constant c is 0, and therefore the trace map is not affine. The problem here is that the probability of exiting the loop $c \to c$ is 0. This can potentially be remedied by imposing a suitable guardedness discipline, ensuring productivity of loops, but the following example, adapted from [35, Example 20], shows that even then a.s.t. can fail for probabilistic De Simone specifications, at least for infinite signatures. Take $\Sigma X = \{c_n \mid n \in \mathbb{N}\}$, and

$$\frac{}{c_n \xrightarrow{a/(2^n+2)^{-1}} *} \ (n \in \mathbb{N}) \qquad \frac{}{c_n \xrightarrow{a/(1-(2^n+2)^{-1})} c_{n+1}} \ (n \in \mathbb{N})$$

It follows that the total weight $\mathsf{tr}(c_0)$ assigns to traces is $1/2$, hence the system is not a.s.t.

8 Conclusion and Future Work

We have developed a theory of compositional trace semantics for process specifications, emerging at the generality of *abstract GSOS* [44] along with *coalgebraic trace semantics* [25]. One important message of our paper is that "vanilla" abstract GSOS is more powerful than it seems: unlike existing categorical approaches to the subject (see Related Work), our results are based on a slightly refined version of Turi and Plotkin's original results, applied to our novel notion of De Simone laws over Kleisli categories. We have illustrated the usefulness of this abstract framework by deriving from it a novel De Simone-type format for probabilistic LTS. Various subtleties of the format (e.g. regarding almost sure termination) would be hard to uncover from scratch, but are naturally explained and motivated by the categorical setup.

Within the probabilistic setting, our compositionality result is currently limited to almost-surely terminating systems. In future work, we are planning to analyse this issue through the lens of (coalgebraic) *infinite trace semantics* [26], which remedies "probability leakage", caused by non-termination. Kock's synthetic approach to measurability and probability [33] suggests that our framework may be applicable to more sophisticated probabilistic settings, via suitable additive monads. An orthogonal direction is to reformulate our framework internally to a *Markov category* [15, 20], instead of a Kleisli category of a monad.

We also aim to apply our framework to computational effects beyond probabilities and cover other features, such as states or higher-order behaviour. These have been addressed recently from the perspective of abstract GSOS [21, 23, 22] under (weak) bisimilarity, but a

corresponding abstract treatment of trace semantics is still missing. Another direction are congruence formats for notions of *trace distance* in lieu of *trace equivalence*. This will require departing from final coalgebra semantics in favour of a fibrational approach [6, 12, 45].

Finally, we briefly discuss an alternative route to coalgebraic trace equivalence that we have explored: generalized determinization [41]. The idea of this setting is to go from a coalgebra $X \to \mathcal{P}(L \times X) \cong (\mathcal{P}X)^L$ to a "determinized" coalgebra $\mathcal{P}X \to (\mathcal{P}X)^L$ (using the distributive law of the monad $\mathcal{P}$ over the functor $(-)^L$ and the monad multiplication). Technically, this amounts to considering GSOS laws (of polynomial functors over $(-)^L$) and their operational models in the Eilenberg-Moore category $\mathsf{EM}(\mathcal{P})$ and subsequently apply the Turi-Plotkin theory in that category. This looks appealing but, in general, the GSOS law corresponding to a De Simone specification does not yield a morphism in $\mathsf{EM}(\mathcal{P})$, that is, a lattice morphism.

A specification forming a lattice morphism means that for any binary operation f, if $\mathsf{f}(t_1, t_2) \xrightarrow{a} t_3$ and $\mathsf{f}(u_1, u_2) \xrightarrow{a} u_3$ then $\mathsf{f}(t_1, t_2) \vee \mathsf{f}(u_1, u_2) \xrightarrow{a} t_3 \vee u_3$. Also, the fact that Σ lifts entails that $\mathsf{f}(t_1, t_2) \vee \mathsf{f}(u_1, u_2) = \mathsf{f}(t_1 \vee u_1, t_2 \vee u_2)$. Now suppose we have $L = \{a\}$, terms $t_1 \xrightarrow{a}, t_2 \xrightarrow{a}, u_1 \xrightarrow{a}, u_2 \xrightarrow{a} v$, and a binary operation f with the unique (De Simone) rule

$$\frac{x_2 \xrightarrow{a} y_2}{\mathsf{f}(x_1, x_2) \xrightarrow{a} x_1}.$$

Then $\mathsf{f}(t_1, t_2) \xrightarrow{a}$, $\mathsf{f}(u_1, u_2) \xrightarrow{a} u_1$ but $\mathsf{f}(t_1 \vee u_1, t_2 \vee u_2) \xrightarrow{a} t_1 \vee u_1$, contradicting the above properties.

This shows that the De Simone rule format is not compatible with the approach via generalized determinization and it is unclear whether there is any such format altogether.

References

1 Faris Abou-Saleh. *A coalgebraic semantics for imperative programming languages*. PhD thesis, Imperial College London, UK, 2014. URL: http://hdl.handle.net/10044/1/13693.

2 Faris Abou-Saleh and Dirk Pattinson. Towards effects in mathematical operational semantics. In Michael W. Mislove and Joël Ouaknine, editors, *Mathematical Foundations of Programming Semantics, MFPS 2011*, volume 276 of *ENTCS*, pages 81–104. Elsevier, 2011. doi:10.1016/j.entcs.2011.09.016.

3 Faris Abou-Saleh and Dirk Pattinson. Comodels and effects in mathematical operational semantics. In Frank Pfenning, editor, *Foundations of Software Science and Computation Structures - 16th International Conference, FOSSACS 2013, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2013, Rome, Italy, March 16-24, 2013. Proceedings*, volume 7794 of *Lecture Notes in Computer Science*, pages 129–144. Springer, 2013. doi:10.1007/978-3-642-37075-5_9.

4 Luca Aceto, Wan J. Fokkink, and Chris Verhoef. Structural operational semantics. In Jan A. Bergstra, Alban Ponse, and Scott A. Smolka, editors, *Handbook of Process Algebra*, pages 197–292. North-Holland / Elsevier, 2001. doi:10.1016/b978-044482830-9/50021-7.

5 Giorgio Bacci and Marino Miculan. Structural operational semantics for continuous state probabilistic processes. In Dirk Pattinson and Lutz Schröder, editors, *Coalgebraic Methods in Computer Science*, pages 71–89, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. doi:10.1007/978-3-642-32784-1_5.

6 Paolo Baldan, Filippo Bonchi, Henning Kerstan, and Barbara König. Coalgebraic behavioral metrics. *Log. Methods Comput. Sci.*, 14(3), 2018. doi:10.23638/LMCS-14(3:20)2018.

7 Michael Barr. Coequalizers and free triples. *Math. Z.*, 116:307–322, 1970.

8 Falk Bartels. GSOS for probabilistic transition systems. In Lawrence S. Moss, editor, *Coalgebraic Methods in Computer Science, CMCS 2002, Satellite Event of ETAPS 2002, Grenoble, France, April 6-7, 2002*, volume 65 of *Electronic Notes in Theoretical Computer Science*, pages 29–53. Elsevier, 2002. doi:10.1016/S1571-0661(04)80358-X.

9 Falk Bartels. *On generalised coinduction and probabilistic specification formats: Distributive laws in coalgebraic modelling*. PhD thesis, Vrije Universiteit Amsterdam, 2004.

10 Bard Bloom. When is partial trace equivalence adequate? *Formal Aspects of Computing*, 6(3):317–338, 1994. doi:10.1007/BF01215409.

11 Bard Bloom, Sorin Istrail, and Albert R. Meyer. Bisimulation can't be traced. *J. ACM*, 42(1):232–268, 1995. doi:10.1145/200836.200876.

12 Filippo Bonchi, Barbara König, and Daniela Petrisan. Up-to techniques for behavioural metrics via fibrations. In Sven Schewe and Lijun Zhang, editors, *29th International Conference on Concurrency Theory, CONCUR 2018, Beijing, China, September 4-7, 2018*, LIPIcs, pages 17:1–17:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CONCUR.2018.17.

13 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.

14 Valentina Castiglioni, Ruggero Lanotte, and Simone Tini. Back to the format: A survey on sos for probabilistic processes. *Journal of Logical and Algebraic Methods in Programming*, 137:100929, 2024. doi:10.1016/j.jlamp.2023.100929.

15 Kenta Cho and Bart Jacobs. Disintegration and Bayesian inversion via string diagrams. *Mathematical Structures in Computer Science*, 29, 2019. arXiv:1709.00322. doi:10.1017/S0960129518000488.

16 Dion Coumans and Bart Jacobs. Scalars, monads, and categories. In *Quantum Physics and Linguistics - A Compositional, Diagrammatic Discourse*, pages 184–216. Oxford University Press, 2013. doi:10.1093/ACPROF:OSO/9780199646296.003.0007.

17 R. de Simone. Higher-level synchronising devices in meije-sccs. *Theoretical Computer Science*, 37(3):245–267, 1985. doi:10.1016/0304-3975(85)90093-3.

18 Manfred Droste, Werner Kuich, and Heiko Vogler, editors. *Handbook of Weighted Automata*. Springer, 2009.

19 Florian Frank, Stefan Milius, and Henning Urbat. Coalgebraic semantics for nominal automata. In *Coalgebraic Methods in Computer Science - 16th IFIP WG 1.3 International Workshop, CMCS 2022, Colocated with ETAPS 2022, Munich, Germany, April 2-3, 2022, Proceedings*, volume 13225 of *Lecture Notes in Computer Science*, pages 45–66. Springer, 2022. doi:10.1007/978-3-031-10736-8_3.

20 Tobias Fritz. A synthetic approach to Markov kernels, conditional independence and theorems on sufficient statistics. *Advances in Mathematics*, 370, 2020. arXiv:1908.07021. doi:10.1016/j.aim.2020.107239.

21 Sergey Goncharov, Stefan Milius, Lutz Schröder, Stelios Tsampas, and Henning Urbat. Stateful structural operational semantics. In Amy P. Felty, editor, *7th International Conference on Formal Structures for Computation and Deduction, FSCD'22*, volume 228 of *LIPIcs*, pages 30:1–30:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.FSCD.2022.30.

22 Sergey Goncharov, Stefan Milius, Lutz Schröder, Stelios Tsampas, and Henning Urbat. Towards a higher-order mathematical operational semantics. In *50th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL 2023)*, volume 7 of *Proc. ACM Program. Lang.* ACM, 2023. doi:10.1145/3571215.

23 Sergey Goncharov, Stefan Milius, Lutz Schröder, Stelios Tsampas, and Henning Urbat. Bialgebraic reasoning on stateful languages, 2025. To appear in Proc. ICFP 2025. doi:10.48550/arXiv.2503.10955.

24 Jan Friso Groote and Frits W. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Inf. Comput.*, 100(2):202–260, 1992. doi:10.1016/0890-5401(92)90013-6.

25 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.

26 Bart Jacobs. Trace semantics for coalgebras. *Electr. Notes Theor. Comput. Sci.*, 106:167–184, 2004. doi:10.1016/j.entcs.2004.02.031.

27 Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*, volume 59 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2016. doi:10.1017/CBO9781316823187.

28 Bartek Klin. Bialgebraic methods and modal logic in structural operational semantics. *Inf. Comput.*, 207(2):237–257, 2009. doi:10.1016/J.IC.2007.10.006.

29 Bartek Klin. Structural operational semantics and modal logic, revisited. In *Proceedings of the Tenth Workshop on Coalgebraic Methods in Computer Science, CMCS@ETAPS 2010, Paphos, Cyprus, March 26-28, 2010*, number 2 in Electronic Notes in Theoretical Computer Science, pages 155–175. Elsevier, 2010. doi:10.1016/J.ENTCS.2010.07.019.

30 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.

31 Bartek Klin and Vladimiro Sassone. Structural operational semantics for stochastic process calculi. In Roberto M. Amadio, editor, *11th International Conference Foundations of Software Science and Computational Structures, FOSSACS'08*, volume 4962 of *LNCS*, pages 428–442. Springer, 2008. doi:10.1007/978-3-540-78499-9_30.

32 Bartek Klin and Vladimiro Sassone. Structural operational semantics for stochastic and weighted transition systems. *Inf. Comput.*, 227:58–83, 2013. doi:10.1016/J.IC.2013.04.001.

33 Anders Kock. Commutative monads as a theory of distributions. *Theory and Applications of Categories*, 26(4):97–131, 2012. URL: http://www.tac.mta.ca/tac/volumes/26/4/26-04abs.html.

34 Joachim Lambek. A fixpoint theorem for complete categories. *Math. Z.*, 103:151–161, 1968.

35 Paul Blain Levy and Sergey Goncharov. Coinductive resumption monads: Guarded iterative and guarded elgot. In *Proc. 8rd international conference on Algebra and coalgebra in computer science (CALCO 2019)*, LIPIcs. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CALCO.2019.13.

36 Harald Lindner. Affine parts of monads. *Arch. Math.*, 33(1):437–443, December 1979.

37 S. Mac Lane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer, 2 edition, 1978. URL: http://link.springer.com/10.1007/978-1-4757-4721-8.

38 Marino Miculan and Marco Peressotti. Structural operational semantics for non-deterministic processes with quantitative aspects. *Theor. Comput. Sci.*, 655:135–154, 2016. doi:10.1016/j.tcs.2016.01.012.

39 Philip S. Mulry. Lifting theorems for kleisli categories. In *Mathematical Foundations of Programming Semantics, 9th International Conference, New Orleans, LA, USA, April 7-10, 1993, Proceedings*, volume 802 of *Lecture Notes in Computer Science*, pages 304–319. Springer, 1993. doi:10.1007/3-540-58027-1_15.

40 Jan J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theor. Comput. Sci.*, 249(1):3–80, 2000. doi:10.1016/S0304-3975(00)00056-6.

41 Alexandra Silva, Filippo Bonchi, Marcello M. Bonsangue, and Jan J. M. M. Rutten. Generalizing determinization from automata to coalgebras. *Logical Methods in Computer Science*, 9(1), 2013. doi:10.2168/LMCS-9(1:9)2013.

42 Stelios Tsampas, Christian Williams, Andreas Nuyts, Dominique Devriese, and Frank Piessens. Abstract congruence criteria for weak bisimilarity. In Filippo Bonchi and Simon J. Puglisi, editors, *46th International Symposium on Mathematical Foundations of Computer Science, MFCS'21*, volume 202 of *LIPIcs*, pages 88:1–88:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.MFCS.2021.88.

43 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.

44 Daniele Turi and Gordon D. Plotkin. Towards a mathematical operational semantics. In *12th Annual IEEE Symposium on Logic in Computer Science (LICS 1997)*, pages 280–291, 1997. doi:10.1109/LICS.1997.614955.

45 Henning Urbat. Higher-order behavioural conformances via fibrations. *Proc. ACM Program. Lang.*, 10(POPL), January 2026. doi:10.1145/3776697.

46 R. J. van Glabbeek. The linear time - branching time spectrum. In *CONCUR '90 Theories of Concurrency: Unification and Extension*, pages 278–297. Springer, 1990.