Basic Research in Computer Science

Syntactic Formats for Free: An Abstract Approach to Process Equivalence

Bartek Klin
Paweł Sobociński

BRICS Report Series **RS-03-18**

ISSN 0909-0878 **April 2003**

**See back inner page for a list of recent BRICS Report Series publications.
Copies may be obtained by contacting:**

**BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk**

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/03/18/`

Syntactic Formats for Free: An Abstract Approach to Process Equivalence

Bartek Klin Paweł Sobociński
BRICS*
University of Aarhus, Denmark

April, 2003

Abstract

A framework of Plotkin and Turi's, originally aimed at providing an abstract notion of bisimulation, is modified to cover other operational equivalences and preorders. Combined with bialgebraic methods, it yields a technique for the derivation of syntactic formats for transition system specifications which guarantee that various operational preorders are precongruences. The technique is applied to the trace preorder, the completed trace preorder and the failures preorder. In the latter two cases, new syntactic formats guaranteeing precongruence properties are introduced.

1 Introduction

Structural operational semantics [21, 2] is one of the most fundamental frameworks for providing a precise interpretation of programming and specification languages. Due to its flexibility and generality, it has gained much popularity in the theory of concurrent processes. It is usually presented as a labelled transition system (LTS), in which states (sometimes

*Basic Research in Computer Science (`www.brics.dk`), funded by the Danish National Research Foundation.

called processes) are closed terms over some syntactic signature, and transitions are labelled with elements of some fixed set of actions. The transition relation is in turn defined by a transition system specification, i.e., a set of derivation rules.

Many operational equivalences and preorders have been defined on processes. Among these are: bisimulation equivalence [19], simulation preorder, trace preorder, completed trace preorder, failures preorder [13, 23] and many others (for a comprehensive list see [10]). In the case of processes without internal actions, all of the above have been given modal characterisations [10], obtained by considering appropriate subsets of the Hennessy-Milner logic [12].

Reasoning about operational equivalences and preorders is significantly easier when they are congruences (resp. precongruences). This facilitates compositional reasoning and full substitutivity. In general, operational equivalences are not necessarily congruences on processes defined by operational rules. Similarly, operational preorders are not necessarily precongruences. Proofs of such congruence results for given transition system specifications can be quite demanding. This has been an acute problem in the process calculus community and has led to new reasoning approaches, for instance, the notion of barbed congruence [18].

One way to ensure congruential properties is to impose syntactic restrictions (called syntactic formats) on operational rules. Many such formats have been developed. For bisimulation equivalence, the examples are: de Simone format [27], GSOS [8], and ntyft/ntyxt [11], each of these generalising the previous one. For trace equivalence, examples include [31, 5], while several versions of decorated trace preorders have been provided with formats in [6]. For an overview of the subject see [2]. Another approach which generates LTS on which bisimulation is a congruence is the smallest-contexts-as-labels approach [26, 17, 25].

The search for an abstract theory of processes, bisimulation and 'well-behaved' operational semantics has led to development of final coalgebra semantics [24], and — later — of bialgebraic semantics [29, 30] of processes. In these frameworks, the notion of a transition system is parametrised by a notion of behaviour. Bisimulation is modelled abstractly as a span of coalgebra morphisms. The abstract notion specialises to the classical one, indeed, the class of finitely branching labelled transition systems is in 1-1 correspondence with the coalgebras of the functor $\mathcal{P}_{\mathrm{f}}(A \times -)$, and to give a (classical) bisimulation relation is

to give a span of coalgebra morphisms for this functor [3, 24]. Another abstract approach to bisimulation is via open maps [15].

In [29, 30] it was shown how to define operational rules on an abstract level. For abstract transition system specifications defined in this way, bisimulation equivalence (defined abstractly, using spans of coalgebra morphisms) is guaranteed to be a congruence.

At the core of this so-called abstract GSOS is the modelling of a transition system specification as a natural transformation

$$\lambda : \Sigma(\mathrm{id} \times B) \to BT$$

where Σ is the syntactic endofunctor, T is the monad freely generated from Σ, and B is some behaviour endofunctor. In the special case of the behaviour endofunctor $\mathcal{P}_{\mathrm{f}}(A \times -)$, the abstract operational rules specialise to GSOS rules.

The abstract framework which defines bisimulation as a span of coalgebra morphisms is not sufficient for certain purposes [22] and in particular one runs into problems when working with complete partial orders. Recently, another abstract notion of bisimulation, based on topologies (or complete boolean algebras) of tests, has been proposed [20, 28]. Again, for the familiar process behaviour the novel abstract notion is equivalent to the classical one.

In this paper we show that the latter abstract definition of bisimulation can in fact be modified in a structured manner, to yield other known operational equivalences and preorders. We illustrate this approach on trace preorder, completed trace preorder and failures preorder (and respective equivalences). This constitutes another systematic approach to various operational preorders and equivalences, such as those based on testing scenarios and modal logics [10], as well as quantales [1].

Although the framework is general, in this paper we shall concentrate on the category of sets and functions, **Set**. We define the *test-suite fibration* with total category $\mathbf{Set}^*$ having as objects pairs consisting of a set X and a test suite (a subset of $\mathcal{P}X$) over X. We define a way of lifting the abstract-GSOS framework to $\mathbf{Set}^*$ by describing how to lift the syntax functor Σ and the behaviour functor B. By changing how B lifts to $\mathbf{Set}^*$ we alter the specialisation preorder of certain test suites in $\mathbf{Set}^*$. In particular, taking particular liftings which strongly resemble fragments of the Hennessy-Milner logic [12] causes the specialisation preorder to vary between known operational preorders. The abstract framework guaran-

tees precongruence properties. The only hurdle is proving that a particular transition system specification (natural transformation) λ lifts to a natural transformation in $\mathbf{Set}^*$:

$$\lambda : \Sigma^*(\mathrm{id} \times B^*) \to B^*T^*.$$

The consideration of which properties λ must satisfy in order to lift provides us with syntactic sub-formats of GSOS which guarantee precongruence properties for various operational preorders.

In this paper, we illustrate this approach by presenting precongruence formats for the trace preorder, the completed trace preorder and the failures preorder. The format derived for the trace preorder coincides with the well known de Simone format [31] and we include it here as an illustration of the technique on a relatively simple example. The format derived for the completed trace preorder is, to the best of our knowledge, the first such format published. The format derived for the failures preorder is incomparable with the analogous format given in [6].

The structure of the paper is as follows. After §2 of preliminaries, we present the three obtained syntactic formats in §3, together with some examples and counterexamples from literature. The remaining sections are devoted to proving that the presented formats are indeed precongruence formats w.r.t. their respective preorders, and at the same time to illustrating the method of deriving such formats from a given operational preorder. In §4, we recall the basics of bialgebraic semantics. In §5, we present an abstract approach to operational preorders based on the notion of a test suite. In §6, this approach is merged with the bialgebraic framework to yield a general way of checking whether a given operational preorder is a congruence for a given transition system specification. Finally, in §7, we prove that the formats presented in §3 ensure the respective precongruence results. We conclude in §8 by showing possible directions of future work.

Acknowledgements. Most of the contents of §5 and §6 is a modified version of the framework developed (and, unfortunately, not yet published) by Gordon Plotkin [20] and Daniele Turi [28]. We are indebted to Mikkel Nygaard for reading the extended abstract of this report and providing us with many helpful comments. The first author is also grateful to Daniele Turi for introducing him to the subject and for inspiration, and to Gordon Plotkin for discussions and encouragement.

2 Preliminaries

A *labelled transition system* (LTS) is a set P of *processes*, a set A of *actions*, and a *transition relation* $\longrightarrow \subseteq P \times A \times P$. As usual, we write $p \xrightarrow{a} p'$ instead of $\langle p, a, p' \rangle \in \longrightarrow$. An LTS is *finitely branching*, if for every process p there are only finitely many transitions $p \xrightarrow{a} p'$.

Given a set of actions A, three sets of *modal formulae* $\mathcal{F}_{\mathsf{Tr}}$, $\mathcal{F}_{\mathsf{CTr}}$, and $\mathcal{F}_{\mathsf{Fl}}$ are given by the following BNF grammars:

$$\begin{array}{ll} \mathcal{F}_{\mathsf{Tr}} & \phi ::= \top \mid \langle a \rangle \phi \\ \mathcal{F}_{\mathsf{CTr}} & \phi ::= \top \mid \langle a \rangle \phi \mid \tilde{A} \\ \mathcal{F}_{\mathsf{Fl}} & \phi ::= \top \mid \langle a \rangle \phi \mid \tilde{Q} \end{array}$$

where a ranges over A, and Q ranges over subsets of A. Formulae in $\mathcal{F}_{\mathsf{Tr}}$ are called *traces* over A. Formulae in $\mathcal{F}_{\mathsf{CTr}}$ ended with $\tilde{A}$ are called *completed traces*, and formulae in $\mathcal{F}_{\mathsf{Fl}}$ — *failures*.

Given an LTS, the satisfaction relation $\models$ between processes and modal formulae is defined inductively as follows:

$$\begin{array}{lcl} p \models \top & & \\ p \models \langle a \rangle \phi & \iff & p' \models \phi \text{ for some } p' \text{ such that } p \xrightarrow{a} p' \\ p \models \tilde{Q} & \iff & \text{there is no } a \in Q,\ p' \in P \text{ such that } p \xrightarrow{a} p' \end{array}$$

If $p \models \phi$, we will say that ϕ is a trace (completed trace, failure) of p.

Then three operational preorders on the set of processes are defined: the *trace preorder* $\sqsubseteq_{\mathsf{Tr}}$, the *completed trace preorder* $\sqsubseteq_{\mathsf{CTr}}$, and the *failures preorder* $\sqsubseteq_{\mathsf{Fl}}$:

$$p \sqsubseteq_W p' \iff (\forall \phi \in \mathcal{F}_W . p \models \phi \Longrightarrow p' \models \phi)$$

where $\mathsf{W} \in \{\mathsf{Tr}, \mathsf{CTr}, \mathsf{Fl}\}$.

In the context of structural operational semantics, processes are usually closed terms over some signature. A *signature* Σ is a set (also denoted Σ) of *language constructs*, together with an *arity function* $ar : \Sigma \to \mathbb{N}$. For a given set X of *variables*, ΣX is the set of expressions of the form $\mathtt{f}(x_1, \ldots, x_{ar(\mathtt{f})})$, where $\mathtt{f} \in \Sigma$ and $x_i \in X$. In other words,

$$\Sigma X \cong \coprod_{\mathtt{f} \in \Sigma} X^{ar(\mathtt{f})}$$

where $\coprod$ denotes disjoint union. Given a signature Σ and a set X, the set $T_\Sigma X$ of *terms* over Σ with variables X is (isomorphic to) the least fixpoint of the operator

$$\Phi Y = X + \Sigma Y$$

where $+$ denotes disjoint union of sets. When describing terms from $T_\Sigma X$ the injections $\iota_1 : X \to T_\Sigma X$ and $\iota_2 : \Sigma T_\Sigma X \to T_\Sigma X$ will often be omitted, i.e., we will write $\mathtt{f}(x, y)$ rather than $\iota_2(\mathtt{f}(\iota_1(x), \iota_1(y)))$. Also the subscript in $T_\Sigma X$ will be omitted if Σ is irrelevant or clear from the context. Elements of $T\emptyset$ are called *closed terms* over Σ.

For a term $t \in TX$ and a function $\sigma : X \to Y$, $t[\sigma]$ will denote the term in TY resulting from t by simultaneously replacing every $x \in X$ with $\sigma(x)$.

In the following, we assume a fixed, infinite set of variables Ξ, ranged over by $\mathtt{x}_1, \mathtt{x}_2, \ldots, \mathtt{y}_1, \mathtt{y}_2, \ldots$. Terms with variables from Ξ will be typeset $\mathtt{t}$, $\mathtt{t}'$, etc.

Let us fix an arbitrary set of labels A. For a signature Σ, a *positive* Σ-*literal* is an expression $\mathtt{t} \xrightarrow{a} \mathtt{t}'$, and a *negative* Σ-*literal* is an expression $\mathtt{t} \not\xrightarrow{a}$, where $\mathtt{t}, \mathtt{t}' \in T\Xi$ and $a \in A$. A *transition rule* ρ over Σ is an expression $\frac{H}{\alpha}$, where H is a set of Σ-literals and α is a positive Σ-literal. Elements of H are then called *premises* of ρ, and α — the *conclusion* of ρ. The left-hand side and the right-hand side of the conclusion of ρ are called the *source* and the *target* of ρ, respectively.

Similarly, a Σ-*semiliteral* is either a negative Σ-literal, or an expression $\mathtt{t} \xrightarrow{a}$, where $\mathtt{t} \in T\Xi$ and $a \in A$. A positive literal $\mathtt{t} \xrightarrow{a} \mathtt{t}'$ *completes* the semiliteral $\mathtt{t} \xrightarrow{a}$, and we say that a negative literal completes itself.

A *transition system specification* over Σ is a set of transition rules over Σ.

In the following definition assume a fixed signature Σ, and a *finite* set A.

Format 1 (GSOS). A transition system specification R is in GSOS [8] format if every rule $\rho \in R$ is of the form

$$\frac{\left\{ \mathtt{x}_i \xrightarrow{a_{ij}} \mathtt{y}_{ij} \;:\; i \le n,\ j \le m_i \right\} \cup \left\{ \mathtt{x}_i \not\xrightarrow{b_{ik}} \;:\; i \le n, k \le n_i \right\}}{\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{c} \mathtt{t}}$$

with $\mathtt{f} \in \Sigma$ and $n = ar(\mathtt{f})$, such that $\mathtt{x}_i \in \Xi$ and $\mathtt{y}_{ij} \in \Xi$ are all distinct and are the only variables that occur in ρ. In the following, we will

consider only *image finite* GSOS specifications, i.e. those with finitely many rules for each construct $\mathtt{f} \in \Sigma$ and action $c \in A$.

Given a transition system specification R in GSOS format, one defines a notion of a provable positive literal in a straightforward way. The set of all provable literals forms a finitely branching LTS with closed terms over Σ as processes, and with positive closed literals as transitions (for details, see [2]).

An operational preorder $\sqsubseteq$ is a *precongruence* with respect to a transition system specification R, if in the LTS induced by R, for each $\mathtt{f} \in \Sigma$ with arity n, if $t_1 \sqsubseteq t'_1, \ldots, t_n \sqsubseteq t'_n$, then $\mathtt{f}(t_1, \ldots, t_n) \sqsubseteq \mathtt{f}(t'_1, \ldots, t'_n)$.

The examples in §3 are based on basic process algebra **BPA**. Assuming a finite set A of actions, its syntax Σ is defined by the BNF grammar

$$t ::= \mathtt{0} \mid \alpha t \mid t{+}t$$

and the transition system specification **BPA** over Σ is a collection of rules

$$\frac{}{\alpha\mathtt{x} \xrightarrow{\alpha} \mathtt{x}} \qquad \frac{\mathtt{x} \xrightarrow{\alpha} \mathtt{x}'}{\mathtt{x}+\mathtt{y} \xrightarrow{\alpha} \mathtt{x}'} \qquad \frac{\mathtt{y} \xrightarrow{\alpha} \mathtt{y}'}{\mathtt{x}+\mathtt{y} \xrightarrow{\alpha} \mathtt{y}'}$$

where α ranges over A. When presenting terms over the above syntax, the trailing $\mathtt{0}$'s will be omitted. It is easy to see that **BPA** is in the GSOS format.

3 Precongruence Formats

In this section we introduce the syntactic formats derived using the framework described in the latter parts of the paper. The precongruence properties of these formats are formally stated in §7.

Format 2 (Tr-format). A set of GSOS rules R is in Tr-*format*, if for each $\rho \in R$:

- all premises of ρ are positive,
- no variable occurs more than once in the left-hand sides of premises and in the target.

It is easy to see that this format coincides with the well-known de Simone format [27]. The fact that this syntactic format ensures the trace preorder to be a precongruence was first proved in [31].

We proceed to define an analogous syntactic format for the completed trace preorder.

Definition 1 (CTr-testing set). A CTr-*testing set* over a set of variables $\{\mathtt{x}_1, \ldots, \mathtt{x}_n\}$ is a set of semiliterals P of the form

$$P = \left\{ \mathtt{x}_i \overset{a_i}{\not\longrightarrow} \; : \; i \in I \right\} \cup \left\{ \mathtt{x}_i \xrightarrow{a} \; : \; i \in J, a \in A \right\}$$

where $I, J \subseteq \{1, \ldots, n\}$.

Format 3 (CTr-format). A set of GSOS rules R is in CTr-*format*, if

1. For each rule $\rho \in R$:
   - if ρ has a negative premise $\mathtt{x} \overset{a}{\not\longrightarrow}$, than for every label $b \in A$, ρ has also the negative premise $\mathtt{x} \overset{b}{\not\longrightarrow}$,
   - no variable occurs more than once in the target of ρ,
   - no variable occurs simultaneously in the left-hand side of a premise and in the target of ρ,
   - no variable occurs simultaneously in the left-hand side of a positive premise and in the left-hand side of any other premise of ρ.
2. For each construct $\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n)$ of the language, there exists a sequence $P_1, \ldots, P_k$ of CTr-testing sets over $\{\mathtt{x}_1, \ldots, \mathtt{x}_n\}$, such that
   - For every (possibly renamed) rule $\rho \in R$ with source $\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n)$ there exists a sequence $p_1, \ldots p_k$ of semiliterals from $P_1, \ldots, P_k$ respectively, such that for every $i \in \{1, \ldots, k\}$ there exists a premise r of ρ such that r completes p_i.
   - For every sequence $p_1, \ldots, p_k$ of semiliterals from $P_1, \ldots, P_k$ respectively, there exists a (possibly renamed) rule $\rho \in R$ with source $\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n)$ such that for each premise r of ρ there exists an $i \in \{1, \ldots, k\}$ such that r completes p_i.

Note, in particular, that if $k = 0$ then the first part of condition 2 above is always true. Also, if $k = 1$ and $P_1 = \emptyset$, then the second part of condition 2 is always true.

Proposition 2. BPA is in CTr-format.

Proof. It is clear that all rules of **BPA** satisfy condition 1 of CTr-format. For condition 2, consider a language construct $a-$ corresponding to $a \in A$. Take $k = 0$ and check that condition 2 holds. For the binary construct $+$, take $k = 1$ and $P_1 = \{ \mathtt{x} \xrightarrow{a} \ : a \in A \} \cup \{ \mathtt{y} \xrightarrow{a} \ : a \in A \}$. □

The following example is taken from [2]. Assume $A = \{a, b\}$, and extend **BPA** with an operational rule for the so-called *encapsulation operator* $\partial_{\{b\}}$:

$$\frac{\mathtt{x} \xrightarrow{a} \mathtt{y}}{\partial_{\{b\}}(\mathtt{x}) \xrightarrow{a} \partial_{\{b\}}(\mathtt{y})}$$

Then it is easy to check that $aa + ab \sim_{\mathsf{CTr}} a(a + b)$ but that $\partial_{\{b\}}(aa + ab) \not\sqsubseteq_{\mathsf{CTr}} \partial_{\{b\}}(a(a + b))$.

Another example of an operational construct that is not well behaved with respect to completed traces is the *synchronous composition*, as shown in [31]. Here, we add the rules

$$\frac{\mathtt{x} \xrightarrow{\alpha} \mathtt{x}' \quad \mathtt{y} \xrightarrow{\alpha} \mathtt{y}'}{\mathtt{x} \times \mathtt{y} \xrightarrow{\alpha} \mathtt{x}' \times \mathtt{y}'}$$

where α ranges over $A = \{a, b\}$. Here it is easy to see that $aa \times (aa + ab) \not\sqsubseteq_{\mathsf{CTr}} aa \times a(a + b)$.

These two examples have led the authors of [2] to speculate that one cannot hope for a general syntactic congruence format for completed trace equivalence.

Proposition 3. The semantics for the encapsulation operator ∂ and the synchronous composition $\times$ are not in CTr-format.

Proof. Both semantics fail to satisfy condition 2 of CTr-format.

For the encapsulation operator, assume a sequence $P_1, \ldots, P_k$ of CTr-testing sets over $\{\mathtt{x}\}$. From the first part of condition 2, we have $\mathtt{x} \xrightarrow{a} \in P_i$ for $1 \leq i \leq k$. This, by definition of CTr-testing set, means that also $\mathtt{x} \xrightarrow{b} \in P_i$ for $1 \leq i \leq k$. Now take $p_i = \mathtt{x} \xrightarrow{b}$ for $1 \leq i \leq k$ and see that no rule satisfies the second part of condition 2.

For the synchronous composition operator, assume a sequence $P_1, \ldots, P_k$ of CTr-testing sets over $\{\mathtt{x}, \mathtt{y}\}$. From the first part of condition 2, we have that for each $1 \leq i \leq k$, $\mathtt{x} \xrightarrow{a} \in P_i$ or $\mathtt{y} \xrightarrow{a} \in P_i$. By definition

of a CTr-testing set, this means that for each $1 \leq i \leq k$, $\mathtt{x} \xrightarrow{a} \in P_i$ or $\mathtt{y} \xrightarrow{b} \in P_i$. Now for each $1 \leq i \leq k$, take p_i to be $\mathtt{x} \xrightarrow{a}$ if $\mathtt{x} \xrightarrow{a} \in P_i$, and $\mathtt{y} \xrightarrow{b}$ otherwise. It is easy to see that no rule satisfies the second part of condition 2 for this sequence of p_i. □

For a non-trivial example of a transition system specification in CTr-format, extend **BPA** with *sequential composition*, defined by rules

$$\frac{\mathtt{x} \xrightarrow{\alpha} \mathtt{x}'}{\mathtt{x};\mathtt{y} \xrightarrow{\alpha} \mathtt{x}';\mathtt{y}} \qquad \frac{\mathtt{x} \not\xrightarrow{a} \text{ for all } a \in A \quad \mathtt{y} \xrightarrow{\alpha} \mathtt{y}'}{\mathtt{x};\mathtt{y} \xrightarrow{\alpha} \mathtt{y}'}$$

where α ranges over A.

Proposition 4. BPA extended with sequential composition is in CTr-format.

Proof. Condition 1 of CTr-format is checked easily. For condition 2, it is enough to check it for the sequential composition operator. Take $k = |A| + 1$, and

$$\begin{aligned} P_i &= \{\, \mathtt{x} \xrightarrow{a} \,:\, a \in A \,\} \cup \{\mathtt{x} \not\xrightarrow{a_i}\} && \text{for } 1 \leq i < k \\ P_k &= \{\, \mathtt{x} \xrightarrow{a} \,:\, a \in A \,\} \cup \{\, \mathtt{y} \xrightarrow{a} \,:\, a \in A \,\} \end{aligned}$$

It is straightforward to check that both parts of condition 2 hold for this choice of P_i. □

We proceed to define a precongruence syntactic format for the failures preorder.

Definition 5 (FI-testing set). An FI-*testing set* over a set of variables $\{\mathtt{x}_1, \ldots, \mathtt{x}_n\}$ is a set of semiliterals P of the form

$$P = \left\{\, \mathtt{x}_i \not\xrightarrow{a_i} \,:\, i \in I \,\right\} \cup \left\{\, \mathtt{x}_i \xrightarrow{b_{ij}} \,:\, 1 \leq i \leq n, 1 \leq j \leq m_i \,\right\}$$

(where $I \subseteq \{1, \ldots, n\}$, $m_i \in \mathbb{N}$), such that for any labels $a, b \in A$, if $\mathtt{x}_i \xrightarrow{a} \in P$ and $\mathtt{x}_i \not\xrightarrow{b} \in P$ then $\mathtt{x}_i \xrightarrow{b} \in P$.

Format 4 (Failures Format). A set of GSOS rules R is in FI-*format*, if

1. For each rule $\rho \in R$:

- no variable occurs more than once in the target of ρ,
- no variable occurs simultaneously in the left-hand side of a premise and in the target of ρ,
- no variable occurs simultaneously in the left-hand side of a positive premise and in the left-hand side of any other premise of ρ.

2. For each construct $\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n)$ of the language, and for each set of labels $Q \subseteq A$, there exists a sequence $P_1, \ldots, P_k$ of FI-testing sets over $\{\mathtt{x}_1, \ldots, \mathtt{x}_n\}$, such that

   - For every (possibly renamed) rule $\rho \in R$ with the conclusion $f(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{a} \mathtt{t}$ with $a \in Q$ and an arbitrary $\mathtt{t}$, there exists a sequence $p_1, \ldots, p_k$ of semiliterals from $P_1, \ldots, P_k$ respectively, such that for every $i \in \{1, \ldots, k\}$ there exists a premise r of ρ such that r completes p_i.
   - For every sequence $p_1, \ldots, p_k$ of semiliterals from $P_1, \ldots, P_k$ respectively, there exist a label $a \in Q$, a term $\mathtt{t}$, and a (possibly renamed) rule $\rho \in R$ with the conclusion $f(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{a} \mathtt{t}$ such that for each premise r of ρ there exists an $i \in \{1, \ldots, k\}$ such that r completes p_i.

Proposition 6. **BPA** is in FI-format.

Proof. Again, clearly all rules of **BPA** satisfy condition 1 of FI-format. For condition 2, consider a language construct $a-$ corresponding to $a \in A$. For any $Q \subseteq A$, if $a \in Q$ then take $k = 0$ and check that condition 2 holds. Similarly, if $a \notin Q$, take $k = 1$ and $P_1 = \emptyset$. For the binary construct $+$, given $Q \subseteq A$, take $k = 1$ and

$$P_1 = \{\, \mathtt{x} \xrightarrow{a} \;:\; a \in Q \,\} \cup \{\, \mathtt{y} \xrightarrow{a} \;:\; a \in Q \,\}$$

and check that condition 2 holds. □

In [7] it was shown that the failures preorder is not a precongruence for **BPA** extended with sequential composition.

Proposition 7. If A contains at least two different labels a, b, then **BPA** extended with sequential composition is not in FI-format.

Proof. Condition 2 of FI-format fails for the sequential composition operator. Indeed, take $Q = \{a\}$ and assume a sequence $P_1, P_2, \ldots, P_k$ of FI-testing sets over $\{\mathtt{x}, \mathtt{y}\}$. Consider the first part of condition 2 applied to the first rule for the sequential operator instantiated with $\alpha = a$. It is easy to see that $\mathtt{x} \xrightarrow{a} \in P_i$ for all $1 \leq i \leq k$. Similarly, applying the same condition to the second rule (instantiated with $\alpha = a$), one sees that for all $1 \leq i \leq k$, either $\mathtt{y} \xrightarrow{a} \in P_i$ or $\mathtt{x} \not\xrightarrow{b} \in P_i$ for some $b \in A$. This means that there exists a sequence $p_1, \ldots, p_k$ such that $p_i \in P_i$ and $p_i \neq \mathtt{x} \xrightarrow{a}$ for all $1 \leq i \leq k$.

Now from the second part of condition 2 applied to this sequence it follows that for some $1 \leq i \leq k$, $p_i = \mathtt{x} \not\xrightarrow{b}$ for some $b \neq a$. This means that $\mathtt{x} \not\xrightarrow{b} \in P_i$, and since $\mathtt{x} \xrightarrow{a} \in P_i$, by definition of FI-testing set also $\mathtt{x} \xrightarrow{b} \in P_i$.

Now take

$$p'_i = \begin{cases} \mathtt{x} \xrightarrow{b} & \text{if } p_i = \mathtt{x} \not\xrightarrow{b} \\ p_i & \text{otherwise} \end{cases}$$

The second part of condition 2 fails for this sequence. In the first rule (necessarily instantiated with $\alpha = a$), the premise $\mathtt{x} \xrightarrow{a} \mathtt{x}'$ does not complete any p'_i. In the second rule (also necessarily instantiated with $\alpha = a$), the premise $\mathtt{x} \not\xrightarrow{b}$ does not complete any p'_i. □

The FI-format excludes many examples of transition system specifications that behave well with respect to the failures preorder. Many of these examples are covered by the 'failure trace format' introduced in [6]. However, the latter format excludes also some examples covered by FI-format. Indeed, assume $a, b \in A$ and extend **BPA** with two unary constructs $\mathtt{g}, \mathtt{h}$ and operational rules

$$\frac{\mathtt{x} \xrightarrow{\alpha} \mathtt{x}'}{\mathtt{g}(\mathtt{x}) \xrightarrow{\alpha} \mathtt{h}(\mathtt{x}')} \qquad \frac{\mathtt{x} \not\xrightarrow{a}}{\mathtt{h}(\mathtt{x}) \xrightarrow{b} \mathtt{0}}$$

where α ranges over A.

Proposition 8. BPA extended with $\mathtt{g}$ and $\mathtt{h}$ as above, is in FI-format.

Proof. Condition 1 of FI-format is obviously satisfied, and it is enough to check condition 2 for constructs $\mathtt{g}$ and $\mathtt{h}$ only.

For $\mathtt{g}$, for any $Q \subseteq A$ take $k = 1$ and $P_1 = \{\, \mathtt{x} \xrightarrow{a} \;:\; a \in Q \,\}$. It is easy to see that condition 2 holds. For $\mathtt{h}$, for any $Q \subseteq A$, if $b \in Q$ then take

$k = 1$ and $P_1 = \{\mathtt{x} \stackrel{a}{\not\longrightarrow}\}$. If $b \notin Q$, take $k = 1$ and $P_1 = \emptyset$. Condition 2 is also satisfied. □

However, the rules above are not in the 'failure trace format' proposed in [6]. This means that Fl-format is incomparable with that format.

4 An Abstract Approach

In this section we shall recall the foundations needed for the framework described in §5 and §6. First, we briefly recall how LTS can be described as coalgebras for a specific behaviour endofunctor and briefly recall final coalgebra semantics. We then proceed to recall several notions from the abstract approach to operational semantics of Plotkin and Turi [30].

In the following, $\mathcal{P} : \mathbf{Set} \to \mathbf{Set}$ will denote the (covariant) powerset functor. The (covariant) finite powerset functor $\mathcal{P}_\mathrm{f} : \mathbf{Set} \to \mathbf{Set}$ takes a set to the set of its *finite* subsets.

The reader is referred to [16] for any unexplained categorical notation used henceforward.

4.1 Coalgebras

There is a bijection between the set of finitely branching LTS over a fixed set of actions A and the coalgebras of the functor $\mathcal{P}_\mathrm{f}(A \times -)$. Indeed, given an LTS $\langle P, A, \longrightarrow \rangle$ let

$$h : P \to \mathcal{P}_\mathrm{f}(A \times P)$$

be defined by $h(p) = \{\, \langle a, p' \rangle \,:\, p \stackrel{a}{\longrightarrow} p' \,\}$.

The functor $\mathcal{P}_\mathrm{f}(A \times -)$ has a final coalgebra $\varphi : S \to \mathcal{P}_\mathrm{f}(A \times S)$, that is, a terminal object in the category of $\mathcal{P}_\mathrm{f}(A \times -)$-coalgebras and $\mathcal{P}_\mathrm{f}(A \times -)$-coalgebra morphisms. The carrier S of this coalgebra may be described as the set of synchronisation trees with edges having labels from A, quotiented by bisimulation [4, 29].

4.2 GSOS is natural

In the following we specialise the framework of [30] to the category **Set** and behaviour functor $\mathcal{P}_{\mathrm{f}}(A \times -)$. Any syntactic signature Σ determines a so-called syntactic endofunctor $\Sigma : \mathbf{Set} \to \mathbf{Set}$ which acts on sets by sending

$$\Sigma : X \mapsto \coprod_{\mathtt{f} \in \Sigma} X^{ar(\mathtt{f})}$$

and the action on functions is the obvious one. The functor Σ freely generates a monad $\langle T, \mu, \eta \rangle : \mathbf{Set} \to \mathbf{Set}$. It turns out that TX is (isomorphic to) the set of all terms over Σ with variables from X.

The following theorem is from [30].

Theorem 9. There is a correspondence between sets of rules in the GSOS format (Format 1) and natural transformations

$$\lambda : \Sigma(\mathrm{id} \times \mathcal{P}_{\mathrm{f}}(A \times -)) \to \mathcal{P}_{\mathrm{f}}(A \times T-)$$

Moreover, the correspondence is 1-1 up to equivalence of sets of rules.

Proof. Here we only show how to construct a natural transformation λ from a set of rules R. Given a function symbol $\mathtt{f} \in \Sigma$ with arity n and a set X, a rule ρ in the GSOS format (Format 1)

$$\frac{\left\{ \mathtt{x}_i \xrightarrow{a_{ij}} \mathtt{y}_{ij} \;:\; i \leq n,\ j \leq m_i \right\} \cup \left\{ \mathtt{x}_i \overset{b_{ik}}{\not\longrightarrow} \;:\; i \leq n, k \leq n_i \right\}}{\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n) \xrightarrow{c} \mathtt{t}}$$

defines a map $\rho_X : (X \times \mathcal{P}(A \times X))^n \to \mathcal{P}_{\mathrm{f}}(A \times TX)$ as follows: $\langle c, t \rangle \in \rho_X \langle x_i, \beta_i \rangle_{i \leq n}$ iff there exists $\sigma : \Xi \to X$ satisfying

1. $\sigma(\mathtt{x_i}) = x_i$
2. $\forall i \leq n \, \forall j \leq m_i \ \langle a_{ij}, \sigma(\mathtt{y_{ij}}) \rangle \in \beta_i$
3. $\forall i \leq n \, \forall k \leq n_i \, \forall x \in X \ \langle b_{ij}, x \rangle \notin \beta_i$
4. $\mathtt{t}[\sigma] = t$

Then given a set R of rules in the GSOS format we can define a function $\lambda_X : \Sigma(X \times \mathcal{P}_{\mathrm{f}}(A \times X)) \to \mathcal{P}_{\mathrm{f}}(A \times TX)$ by defining for each $\mathtt{f} \in \Sigma$ a

function $f_X : (X \times \mathcal{P}_{\mathrm{f}}(A \times X))^n \to \mathcal{P}_{\mathrm{f}}(A \times TX)$ as follows:

$$f_X : \langle x_i, U_i \rangle_{i \leq n} \quad \mapsto \quad \bigcup_{\substack{\rho \in R \\ \rho \text{ a rule for } \mathtt{f}}} \rho_X \langle x_i, \beta_i \rangle_{i \leq n}$$

Finally, λ_X is determined uniquely by the f_Xs since it is a function from a coproduct. □

4.3 λ-models

We shall now recall some central results of [30].

Assume a natural transformation $\lambda : \Sigma(\mathrm{id} \times B) \to BT$. A *$\lambda$-model* is a pair

$$\Sigma X \xrightarrow{h} X \xrightarrow{g} BX$$

which satisfies $g \circ h = Bh^\sharp \circ \lambda_X \circ \Sigma \langle \mathrm{id}, g \rangle$, where $h^\sharp : TX \to X$ is the inductive extension of h. A λ-model morphism between $\Sigma X \xrightarrow{h} X \xrightarrow{g} BX$ and $\Sigma X' \xrightarrow{h'} X' \xrightarrow{g'} BX'$ is a morphism $f : X \to X'$ which is simultaneously a Σ-algebra morphism and a B-coalgebra morphism, ie. $h' \circ \Sigma f = g \circ h$ and $g' \circ f = Bf \circ g$. Let λ-**Mod** denote the category of λ-models and λ-model morphisms.

Theorem 10. Suppose that $\mathbf{C}$ is a category, Σ is an endofunctor which freely generates a monad T and B is an endofunctor which cofreely generates a comonad D. Then the following hold:

1. λ-**Mod** has an initial and final object,
2. the carrier and algebra part of the initial λ-model is the initial Σ-algebra,
3. the carrier and coalgebra part of the final λ-model is the final B-coalgebra and
4. the coalgebra part of the initial λ-model is the so-called *intended operational model* of λ.

Proof. See [30]. In particular, if $\mathbf{C} = \mathbf{Set}$ and $B = \mathcal{P}_{\mathrm{f}}(A \times -)$, then the intended operational model of λ is the LTS generated by the GSOS rules associated to λ. □

If the conditions of Theorem 10 hold and $\psi : \Sigma N \to N$ is the initial Σ-algebra and $\varphi : S \to BS$ is the final B-coalgebra then there exists a Σ-algebra $\delta : \Sigma S \to S$ and a B-coalgebra $\epsilon : N \to BN$ so that $\langle \psi, \epsilon \rangle$ is the initial object of λ-**Mod** and $\langle \delta, \varphi \rangle$ is the final object of λ-**Mod**. Then, by initiality (or finality) there exists a unique λ-model morphism $k : N \to S$ as illustrated below:

$$\begin{array}{ccc} \Sigma N & \xrightarrow{Tk} & \Sigma S \\ \downarrow{\scriptstyle \psi} & & \downarrow{\scriptstyle \delta} \\ N & \xrightarrow{k} & S \\ \downarrow{\scriptstyle \epsilon} & & \downarrow{\scriptstyle \varphi} \\ BN & \xrightarrow[Bk]{} & BS. \end{array}$$

In the following, we shall use the fact that theorem 10 applies to **Set**, Σ and $\mathcal{P}_{\mathrm{f}}(A \times -)$.

5 Process Equivalences from Fibred Functors

In this section, we introduce the central concept of a test suite fibration. This is a modification of the yet unpublished framework [20, 28] due to Plotkin and Turi. In that approach, the test suites (Definition 11) are necessarily *topologies*, that is, they satisfy certain closure properties. We relax this definition and require only that a test suite contains the largest test. This modification allows us to consider operational preorders and equivalences different from bisimulation. Also, the original framework was developed largely for **Cppo**-enriched categories, here we deal primarily with **Set**.

We define $2 = \{\mathtt{tt}, \mathtt{ff}\}$. Given a function $f : X \to Y$ and subsets $V \subseteq X$, $V' \subseteq Y$, we shall use $f(V)$ to denote the set $\{\, y \in Y \,:\, \exists x \in X.\, fx = y \,\}$ and similarly $f^{-1}(V')$ to denote $\{\, x \in X \,:\, fx \in V' \,\}$. Given a set $\tau \subseteq \mathcal{P}X$, the *specialisation preorder* of τ is defined by

$$x \leq_\tau x' \quad \text{iff} \quad \forall V \in \tau.\, x \in V \Rightarrow x' \in V$$

We will sometimes omit the subscript in $\leq_\tau$, where τ will be clear from the context.

For an introduction to fibrations and related terminology, the reader is referred to the first chapter of [14].

Definition 11 (Test suite). A *test* on a set X is a function $V : X \to 2$. We say that an element x *passes a test* V iff $Vx = \mathtt{tt}$. A *test suite* on X is a collection of tests on X which includes the maximal test, that is, the function constant at $\mathtt{tt}$. Let X^* denote the poset of test suites on X ordered by inclusion.

We can define a functor $(-)^* : \mathbf{Set}^{\mathrm{op}} \to \mathbf{Pos}$ which sends a set to the poset of test suites X^* and sends a function $f : X \to Y$ to $f^* : Y^* \to X^*$ defined by

$$f^*\tau' = \{\, V' \circ f \;:\; V' \in \tau' \,\}.$$

Intuitively, we usually think of tests on X as subsets of X. Then f^* is the pre-image operation, taking each test on Y to the test on X which maps to Y via f. This intuition poses no problem since there is a canonical isomorphism between the sets of subsets $V \subseteq X$ and functions $V : X \to 2$. Thus we shall sometimes use set-theoretical notation when talking about tests, that is, we shall make use of notions such as membership, union and cartesian product. Similarly, we shall usually denote the maximal test on X as simply X. We shall, however, be careful to indicate when we are using set-theoretical notation as opposed to similar operations on functions.

Definition 12 (Test suite fibration). A *fibration of test suites* for $(-)^*$ is the fibration obtained using the Grothendieck construction, ie. the total category $\mathbf{Set}^*$ has

- objects: pairs $\langle X, \tau \rangle$ where $X \in \mathbf{Set}$ and $\tau \in X^*$, τ is a test suite.
- arrows: $\langle X, \tau \rangle \xrightarrow{f} \langle X', \tau' \rangle$ iff $f : X \to X'$ and $f^*\tau' \subseteq \tau$.

It is then a standard result that the obvious forgetful functor $U : \mathbf{Set}^* \to \mathbf{Set}$ taking $\langle X, \tau \rangle$ to X is a fibration.

It will be useful to define various operations on test suites τ. Letting $\nabla : 2 + 2 \to 2$ be the codiagonal and $\wedge : 2 \times 2 \to 2$ be logical-and, we let

$$\begin{aligned}
\tau \oplus \tau' &= \{\, \nabla \circ (V + V') \;:\; V \in \tau, V' \in \tau' \,\} \\
\tau \otimes \tau' &= \{\, \wedge \circ (V \times V') \;:\; V \in \tau, V' \in \tau' \,\} \\
\tau \bowtie \tau' &= \{\, V \circ \pi_1 \;:\; V \in \tau \,\} \cup \{\, V' \circ \pi_2 \;:\; V' \in \tau' \,\}.
\end{aligned}$$

It is easy to check that given two test suites, families $\tau \oplus \tau'$, $\tau \otimes \tau'$ and $\tau \bowtie \tau'$ are test suites. Intuitively, given test suites τ and τ' on X and Y, $\tau \oplus \tau'$ is the test suite on $X + Y$ obtained by taking (disjoint) unions of tests from τ on X and τ' on Y, $\tau \otimes \tau'$ is the test suite on $X \times Y$ consisting of tests built by performing a test from τ on X *and simultaneously* performing a test from τ' on Y and accepting when both tests accept; finally, $\tau \bowtie \tau'$ is the test on $X \times Y$ which consists of *either* a test from τ on X *or* a test from τ' on Y.

Proposition 13. The category $\mathbf{Set}^*$ has coproducts and products:

$$\langle X, \tau \rangle + \langle Y, \tau' \rangle = \langle X + Y, \tau \oplus \tau' \rangle$$
$$\langle X, \tau \rangle \times \langle Y, \tau' \rangle = \langle X \times Y, \tau \bowtie \tau' \rangle$$

Proof. For the product, the projections are the projections in $\mathbf{Set}$,

$$\langle X, \tau \rangle \xleftarrow{\pi_1} \langle X \times Y, \tau \bowtie \tau' \rangle \xrightarrow{\pi_2} \langle Y, \tau' \rangle$$

and clearly $\pi_1^* \tau = \{ V \circ \pi_1 : V \in \tau \} \subseteq \tau \bowtie \tau'$ and similarly $\pi_2^* \tau' \subseteq \tau \bowtie \tau'$. The universal property follows from the universal property in $\mathbf{Set}$, indeed, given $\langle Z, \tau'' \rangle$ and morphisms $f : \langle Z, \tau'' \rangle \to \langle X, \tau \rangle$ and $g : \langle Z, \tau'' \rangle \to \langle Y, \tau \rangle$ the morphism induced by the universal property in $\mathbf{Set}$, $\langle f, g \rangle : \langle Z, \tau'' \rangle \to \langle X \times Y, \tau \bowtie \tau' \rangle$ is defined in $\mathbf{Set}^*$:

$$\begin{aligned} \langle f, g \rangle^* (\tau \bowtie \tau') &= \langle f, g \rangle^* (\{ V \circ \pi_1 : V \in \tau \} \cup \{ V' \circ \pi_2 : V' \in \tau' \}) \\ &= \{ V \circ f : V \in \tau \} \cup \{ V' \circ g : V' \in \tau \} \\ &\subseteq \tau'' \cup \tau'' = \tau'' \text{ (as } f \text{ and } g \text{ are morphisms in } \mathbf{Set}^*) \end{aligned}$$

For the coproduct, the injections are similarly the underlying injections

$$\langle X, \tau \rangle \xrightarrow{i_1} \langle X + Y, \tau \oplus \tau' \rangle \xleftarrow{i_2} \langle Y, \tau' \rangle$$

indeed, $i_1^*(\tau \oplus \tau') = \{ \nabla \circ (V + W) \circ i_1 : V \in \tau,\ W \in \tau' \} = \tau$ and similarly $i_2^*(\tau \oplus \tau') = \tau'$. The universal property also follows easily from the one in $\mathbf{Set}$. □

In the following it will be also useful to use the following tensor product on $\mathbf{Set}^*$:

$$\langle X, \tau \rangle \otimes \langle Y, \tau' \rangle = \langle X \times Y, \tau \otimes \tau' \rangle$$

Let $B : \mathbf{Set} \to \mathbf{Set}$ be some behaviour endofunctor. A *lifting* of B to $\mathbf{Set}^*$ is an endofunctor $B^* : \mathbf{Set}^* \to \mathbf{Set}^*$ such that, for some $B_X : X^* \to$

$(BX)^*$ we have $B^* \langle X, \tau \rangle = \langle BX, B_X \tau \rangle$ and $B^* f = Bf$. It turns out that there are many possible choices for B_X giving different liftings of the behaviour endofunctor B to $\mathbf{Set}^*$. One systematic way to construct such liftings is via families of functions from $B2$ to 2. Intuitively, such functions correspond to modalities like those in the Hennessy-Milner logic. In the original framework due to Plotkin and Turi [20, 28] the canonical choice of *all* functions from $B2$ to 2 is taken.

For any X, let $\mathrm{Cl}_X : \mathcal{PP}X \to X^*$ denote a closure operator. We shall only demand that for all $f : X \to Y$ and Z some set of subsets of Y we have $\mathrm{Cl}_X f^* Z = f^* \mathrm{Cl}_Y Z$ (with the obvious extension of the domain of f^* to *all* sets of subsets). Intuitively, a closure operator corresponds to a set of propositional connectives.

Given an arbitrary family W of functions $B2 \to 2$, we define an operator $B_X^W : X^* \to (BX)^*$ as follows:

$$B_X^W(\tau) = \mathrm{Cl}_{BX} \{ w \circ BV \,:\, w \in W \text{ and } V \in \tau \}.$$

We are now in a position to construct a lifting of B to $\mathbf{Set}^*$. Indeed, we let $B^W \langle X, \tau \rangle = \langle BX, B_X^W \tau \rangle$ and $B^W f = Bf$. One needs to check that for any $f : \langle X, \tau \rangle \to \langle X', \tau' \rangle$, $Bf : \langle BX, B_X^W \tau \rangle \to \langle BX', B_{X'}^W \tau' \rangle$ is a morphism in $\mathbf{Set}^*$. Indeed,

$$\begin{aligned}
(Bf)^* B_{X'}^W \tau' &= (Bf)^* \mathrm{Cl}_{BX'} \{ w \circ BV' \,:\, w \in W,\, V' \in \tau' \} \\
&= \mathrm{Cl}_{BX} (Bf)^* \{ w \circ BV' \,:\, w \in W,\, V' \in \tau' \} \\
&= \mathrm{Cl}_{BX} \{ w \circ BV' \circ Bf \,:\, w \in W,\, V' \in \tau' \} \\
&= \mathrm{Cl}_{BX} \{ w \circ B(V' \circ f) \,:\, w \in W,\, V' \in \tau' \} \\
&\subseteq \mathrm{Cl}_{BX} \{ w \circ BV \,:\, w \in W,\, V \in \tau \} \qquad (f^* \tau' \subseteq \tau) \\
&= B_X^W \tau
\end{aligned}$$

Theorem 14. Suppose that $B : \mathbf{Set} \to \mathbf{Set}$ has a final coalgebra

$$\varphi : S \to BS.$$

Then $\varphi : \langle S, M^W \rangle \to \langle BS, B_S^W M^W \rangle$ is a final B^W coalgebra where M^W is the least fixpoint of the operator $\Phi(\tau) = \varphi^* B_S^W \tau$ on S^*.

Proof. First, by definition $\varphi^*(B_S^W M) = M$ and therefore φ is a morphism in $\mathbf{Set}^*$.

Suppose that $h : \langle X, \tau \rangle \to \left\langle BX, B_X^W \tau \right\rangle$ is an arbitrary B^W-coalgebra. Then $h : X \to BX$ is a B-coalgebra and we have a unique $k : X \to S$ such that $Bk \circ h = \varphi \circ k$:

$$\begin{array}{ccc} X & \xrightarrow{k} & S \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle \varphi} \\ BX & \xrightarrow[Bk]{} & BS. \end{array}$$

It remains to verify that k is a morphism in the total category. Thus we have to show that $k^*M \subseteq \tau$. We do this by proving that k^*M is the least prefixpoint of the operator $\Phi'(\tau) = h^* B_X^W \tau$. First notice that

$$\begin{aligned} \Phi'(k^*\tau) &= h^* B_X^W k^* \tau \\ &= \mathrm{Cl}_{BX} \{\, w \circ B(V \circ k) \circ h \,:\, w \in W \text{ and } V \in \tau \,\} \\ &= \mathrm{Cl}_{BX} \{\, w \circ BV \circ Bk \circ h \,:\, w \in W \text{ and } V \in \tau \,\} \\ &= \mathrm{Cl}_{BX} \{\, w \circ BV \circ \varphi \circ k \,:\, w \in W \text{ and } V \in \tau \,\} \qquad \text{(diagram)} \\ &= k^* \varphi^* B_S^W \tau \\ &= k^* \Phi(\tau) \end{aligned}$$

Then

$$\bigcup_{n \in \mathbb{N}} \Phi'^n \emptyset = \bigcup_{n \in \mathbb{N}} \Phi'^n k^* \emptyset = \bigcup_{n \in \mathbb{N}} k^* \Phi^n \emptyset = k^* (\bigcup_{n \in \mathbb{N}} \Phi^n \emptyset) = k^* M$$

We know that since h is a B^W coalgebra that $h^* B_X^W \tau = \Phi'(\tau) \subseteq \tau$. Thus it follows immediately that $k^* M \subseteq \tau$. □

Suppose that $B : \mathbf{Set} \to \mathbf{Set}$ lifts to a functor $B^W : \mathbf{Set}^* \to \mathbf{Set}^*$ with B^W defined as before.

Theorem 15. Take any coalgebra $h : X \to BX$, and let $k : X \to S$ be the unique coalgebra morphism from h to the final B-coalgebra. Then k^*M (where $\langle S, M \rangle$ is the carrier of the final B^W-coalgebra) is the least test suite τ on X such that $h : \langle X, \tau \rangle \to \left\langle BX, B_X^W \tau \right\rangle$ is a morphism in $\mathbf{Set}^*$.

Proof. Immediate from the proof of Theorem 14, where we showed that k^*M was the least prefixpoint of $\Phi'(\tau) = h^* B_X^W \tau$. □

From now on we shall assume a finite set of labels A and confine our attention to the endofunctor $BX = \mathcal{P}_{\mathrm{f}}(A \times X)$ on **Set**.

Assuming $a \in A$ and $Q \subseteq A$, let $w_{\langle a \rangle}, w_{rQ} : B2 \to 2$ denote the functions

$$w_{\langle a \rangle} X = \begin{cases} \mathtt{tt} & \text{if } \langle a, \mathtt{tt} \rangle \in X \\ \mathtt{ff} & \text{otherwise,} \end{cases}$$

$$w_{rQ} X = \begin{cases} \mathtt{tt} & \text{if } \forall a \in Q \, \forall v \in 2 \, \langle a, v \rangle \notin X \\ \mathtt{ff} & \text{otherwise.} \end{cases}$$

In particular, $w_{rA} X = \mathtt{tt}$ iff $X = \emptyset$.

We shall now define three subsets of maps $B2 \to 2$:

- $\mathsf{Tr} = \{ \, w_{\langle a \rangle} \, : \, a \in A \, \}$
- $\mathsf{CTr} = \mathsf{Tr} \cup \{ w_{rA} \}$
- $\mathsf{Fl} = \mathsf{Tr} \cup \{ \, w_{rQ} \, : \, Q \subseteq A \, \}$

The set Tr together with the closure operator $\mathrm{Cl}^{\top}_X(\tau) = \tau \cup \{X\}$, determines B^{Tr}_X for any X and therefore determines a lifting of B to $B^{\mathsf{Tr}} : \mathbf{Set}^* \to \mathbf{Set}^*$.

Similarly, CTr with $\mathrm{Cl}^{\top}$ and and Fl with $\mathrm{Cl}^{\top}$ determine liftings B^{CTr} and B^{Fl} respectively.

The following Theorems 16, 17 and 18 show that the three liftings as defined above cause the specialisation preorders in the final B^{Tr}, B^{CTr} and B^{Fl}-coalgebras to coincide with the trace, the completed trace and the failures preorders. We use these facts to prove Theorem 19 which states that given any B-coalgebra (LTS) $h : X \to \mathcal{P}_{\mathrm{f}}(A \times X)$, the specialisation preorder on k^*M where k is the unique morphism to the final coalgebra given by finality, coincides with the operational preorder corresponding to the choice of the lifting of B.

Theorem 16. In the final B^{Tr}-coalgebra, the specialisation preorder coincides with the trace preorder.

Proof. Let $\varphi : \langle S, M^{\mathsf{Tr}} \rangle \to \langle BS, B^{\mathsf{Tr}}_S M^{\mathsf{Tr}} \rangle$ be the final coalgebra of B^{Tr}, constructed as in Theorem 14.

Let α range over traces over the alphabet A. Let S_α denote the set of synchronisation trees in which α is a possible trace.

We know that $M^{\mathsf{Tr}} = \bigcup_n \Phi^n(\emptyset)$ where $\Phi(\tau) = \varphi^* B_S^\top \tau$. We shall show by induction that

$$\Phi^n(\emptyset) = \{\, S_\alpha \,:\, |\alpha| < n \,\}\,.$$

The proposition holds trivially for $n = 1$ as $\Phi(\emptyset) = \mathrm{Cl}^{\mathsf{Tr}}(\emptyset) = \{S\} = \{S_\epsilon\} = \{\, S_\alpha \,:\, |\alpha| < 1 \,\}$.

Now

$$\begin{aligned}\Phi^{n+1}(\emptyset) &= \varphi^* B_S^{\mathsf{Tr}} \Phi^n(\emptyset) \\ &= \left\{\, w_{\langle a\rangle} \circ \mathcal{P}_{\mathrm{f}}(A \times S_\alpha) \circ \varphi \,:\, a \in A,\ |\alpha| < n \,\right\} \cup \{S\}\end{aligned}$$

using the induction hypothesis. We need to understand the synchronisation trees which pass $w_{\langle a\rangle} \circ \mathcal{P}_{\mathrm{f}}(A \times S_\alpha) \circ \varphi$. Unpacking the definition yields

$$\begin{aligned}(w_{\langle a\rangle} \circ \mathcal{P}(A \times S_\alpha) \circ \varphi)s = \mathtt{tt} &\Leftrightarrow \langle a, \mathtt{tt}\rangle \in (\mathcal{P}(A \times S_\alpha) \circ \varphi)s \\ &\Leftrightarrow \langle a, s'\rangle \in \varphi s,\ s' \in S_\alpha \\ &\Leftrightarrow s \xrightarrow{a} s',\ s' \in S_\alpha \\ &\Leftrightarrow s \in S_{a\alpha}\end{aligned}$$

Clearly we have $\{\, S_\alpha \,:\, |\alpha| < n+1 \,\} = \{\, S_{a\alpha} \,:\, a \in A \text{ and } |\alpha| < n \,\} \cup \{S\}$ and so we have established the property.

Now:

$$\begin{aligned}s \leq s' \text{ iff } &\forall V \in M\,(s \in V \Rightarrow s' \in V) \\ \text{iff } &\forall \alpha\,(s \in S_\alpha \Rightarrow s' \in S_\alpha) \\ \text{iff } &\forall \alpha\,(s \models \alpha \Rightarrow s' \models \alpha) \\ \text{iff } &s \sqsubseteq_{\mathsf{Tr}} s'.\end{aligned}$$

□

Suppose that $Q \subseteq A$ and let $S_{\alpha\tilde{Q}}$ denote the set of synchronisation trees which have $\alpha\tilde{Q}$ as a failure.

Theorem 17. In the final B^{CTr}-coalgebra, the specialisation preorder coincides with the completed trace preorder.

Proof. We shall show by induction that for $n \geq 2$

$$\Phi^n(\emptyset) = \{\, S_\alpha \,:\, |\alpha| < n \,\} \cup \{\, S_{\alpha\tilde{A}} \,:\, |\alpha| < n-1 \,\}$$

Indeed, we have

$$\begin{aligned}\Phi^2(\emptyset) &= \Phi(\{S\}) \\ &= \left\{\, w_{\langle a\rangle} \circ \mathcal{P}_{\mathrm{f}}(A \times S) \circ \varphi \,:\, a \in A \,\right\} \cup \{S\} \cup \{w_{rA} \circ \mathcal{P}_{\mathrm{f}}(A \times S) \circ \varphi\} \\ &= \{\, S_\alpha \,:\, |\alpha| < 2 \,\} \cup S_{\epsilon\tilde{A}}\end{aligned}$$

since $w_{rA} \circ \mathcal{P}(A \times S) \circ \varphi = S_{\epsilon\tilde{A}}$ is the singleton set containing the (equivalence class of) synchronisation tree with one node and no transitions.

Now

$$\begin{aligned}\Phi^{n+1}(\emptyset) &= \Phi(\Phi^n(\emptyset)) \\ &= \Phi(\{\, S_\alpha \,:\, |\alpha| < n \,\} \cup \{\, S_{\alpha\tilde{A}} \,:\, |\alpha| < n-1 \,\}) \qquad \text{(ind. hyp.)} \\ &= \left\{\, w_{\langle a\rangle} \circ \mathcal{P}_{\mathrm{f}}(A \times S_\alpha) \circ \varphi \,:\, a \in A \text{ and } |\alpha| < n \,\right\} \cup S \\ &\quad \cup \left\{\, w_{\langle a\rangle} \circ \mathcal{P}_{\mathrm{f}}(A \times S_{\alpha\tilde{A}}) \circ \varphi \,:\, a \in A \text{ and } |\alpha| < n-1 \,\right\} \\ &\quad \cup \{\, w_{rA} \circ \mathcal{P}_{\mathrm{f}}(A \times S_\alpha) \circ \varphi \,:\, |\alpha| < n \,\} \\ &\quad \cup \{\, w_{rA} \circ \mathcal{P}_{\mathrm{f}}(A \times S_{\alpha\tilde{A}}) \circ \varphi \,:\, |\alpha| < n-1 \,\} \\ &= \{\, S_\alpha \,:\, |\alpha| < n+1 \,\} \cup \{\, S_{a\alpha\tilde{A}} \,:\, a \in A \text{ and } |\alpha| < n-1 \,\} \cup S_{\epsilon\tilde{A}} \\ &= \{\, S_\alpha \,:\, |\alpha| < n+1 \,\} \cup \{\, S_{\alpha\tilde{A}} \,:\, |\alpha| < n \,\}\end{aligned}$$

It follows that:

$$\begin{aligned}s \leq s' &\text{ iff } \forall V \in M\,(s \in V \Rightarrow s' \in V) \\ &\text{iff } \forall \alpha\,(s \in S_\alpha \Rightarrow s' \in S_\alpha) \wedge \forall \alpha\,(s \in S_{\alpha,A} \Rightarrow s' \in S_{\alpha,A}) \\ &\text{iff } \forall \alpha\,(s \models \alpha \Rightarrow s' \models \alpha) \wedge \forall \alpha\,(s \models \alpha\tilde{A} \Rightarrow s' \models \alpha\tilde{A}) \\ &\text{iff } s \sqsubseteq_{\mathsf{CTr}} s'.\end{aligned}$$

□

Theorem 18. In the final B^{Fl}-coalgebra, the specialisation preorder coincides with the failures preorder.

Proof. We shall show by induction for $n \geq 2$ that

$$\Phi^n(\emptyset) = \{\, S_\alpha \,:\, |\alpha| < n \,\} \cup \left\{\, S_{\alpha\tilde{Q}} \,:\, |\alpha| < n-1 \text{ and } Q \subseteq A \,\right\}$$

We have

$$\begin{aligned}\Phi^2(\emptyset) &= \Phi(\{S\}) \\ &= \left\{ w_{\langle a\rangle} \circ \mathcal{P}_{\mathrm{f}}(A \times S) \circ \varphi \,:\, a \in A \right\} \cup \{S\} \\ &\quad \cup \left\{ w_{rQ} \circ \mathcal{P}_{\mathrm{f}}(A \times S) \circ \varphi \,:\, a \in A \text{ and } Q \subseteq A \right\} \\ &= \left\{ S_\alpha \,:\, |\alpha| < 2 \right\} \cup \left\{ S_{\epsilon\tilde{Q}} \,:\, Q \subseteq A \right\}\end{aligned}$$

Now,

$$\begin{aligned}\Phi^{n+1}(\emptyset) &= \Phi(\Phi^n(\emptyset)) \\ &= \Phi(\left\{ S_\alpha \,:\, |\alpha| < n \right\} \cup \left\{ S_{\alpha\tilde{Q}} \,:\, |\alpha| < n-1 \text{ and } Q \subseteq A \right\}) \\ &= \left\{ w_{\langle a\rangle} \circ \mathcal{P}_{\mathrm{f}}(A \times S_\alpha) \circ \varphi \,:\, a \in A \text{ and } |\alpha| < n \right\} \cup S \\ &\quad \cup \left\{ w_{\langle a\rangle} \circ \mathcal{P}_{\mathrm{f}}(A \times S_{\alpha\tilde{Q}}) \circ \varphi \,:\, a \in A, |\alpha| < n-1 \text{ and } Q \subseteq A \right\} \\ &\quad \cup \left\{ w_{rQ} \circ \mathcal{P}_{\mathrm{f}}(A \times S_\alpha) \circ \varphi \,:\, |\alpha| < n \text{ and } Q \subseteq A \right\} \\ &\quad \cup \left\{ w_{rQ} \circ \mathcal{P}_{\mathrm{f}}(A \times S_{\alpha\tilde{Q}'}) \circ \varphi \,:\, |\alpha| < n-1 \text{ and } Q, Q' \subseteq A \right\} \\ &= \left\{ S_\alpha \,:\, |\alpha| < n+1 \right\} \\ &\quad \cup \left\{ S_{a\alpha\tilde{Q}} \,:\, a \in A,\ |\alpha| < n-1 \text{ and } Q \subseteq A \right\} \cup \left\{ S_{\epsilon\tilde{Q}} \,:\, Q \subseteq A \right\} \\ &= \left\{ S_\alpha \,:\, |\alpha| < n+1 \right\} \cup \left\{ S_{\alpha\tilde{Q}} \,:\, |\alpha| < n \text{ and } Q \subseteq A \right\}\end{aligned}$$

Noticing that $S_\alpha = S_{\alpha\tilde{\emptyset}}$ we can write

$$M^{\mathsf{FI}} = \bigcup_n \Phi^n(\emptyset) = \bigcup_n \left\{ S_{\alpha\tilde{Q}} \,:\, |\alpha| < n \text{ and } Q \subseteq A. \right\}$$

It is easy to see that:

$$\begin{aligned}s \leq s' &\text{ iff } \forall V \in M\,(s \in V \Rightarrow s' \in V) \\ &\text{iff } \forall \alpha, Q \subseteq A\,(s \in S_{\alpha\tilde{Q}} \Rightarrow s' \in S_{\alpha\tilde{Q}}) \\ &\text{iff } \forall \alpha, Q \subseteq A\,(s \models \alpha\tilde{Q} \Rightarrow s' \models \alpha\tilde{Q}) \\ &\text{iff } s \sqsubseteq_{\mathsf{FI}} s'.\end{aligned}$$

□

Theorem 19. Suppose that $h : X \to \mathcal{P}(A \times X)$ is a coalgebra (LTS), $\varphi : S \to BS$ is the final B-coalgebra and that $k : X \to S$ is the unique morphism given by finality. Then $x \leq x'$ according to the specialisation preorder of

- $k^*(M^{\mathsf{Tr}})$ iff $x \sqsubseteq_{\mathsf{Tr}} x'$
- $k^*(M^{\mathsf{CTr}})$ iff $x \sqsubseteq_{\mathsf{CTr}} x'$
- $k^*(M^{\mathsf{Fl}})$ iff $x \sqsubseteq_{\mathsf{Fl}} x'$.

Proof. We shall prove the first item, the other two are similar. We have

$$\begin{aligned} x \leq x' &\Leftrightarrow \forall V \in M^{\mathsf{Tr}}\,(x \in h^*V \Rightarrow x' \in h^*V) \\ &\Leftrightarrow \forall V \in M^{\mathsf{Tr}}\,(hx \in V \Rightarrow hx' \in V) \\ &\Leftrightarrow kx \sqsubseteq_{\mathsf{Tr}} kx' && \text{(Theorem 16)} \\ &\Leftrightarrow x \sqsubseteq_{\mathsf{Tr}} x' \end{aligned}$$

where the last step follows from the fact that k is a coalgebra morphism and therefore $\{\, \langle x, kx\rangle \,:\, x \in X \,\}$ is a bisimulation [24]. □

6 Application: Congruence Formats from Bialgebras

To lift the bialgebraic framework to the total category $\mathbf{Set}^*$, we need a way to lift the syntactic and the behaviour functors together with the natural transformation λ. Various ways to lift the behaviour B were shown in the previous section, now we proceed to show a lifting of the syntactic functor.

Given a syntactic endofunctor Σ on $\mathbf{Set}$:

$$\Sigma X = \coprod_{i=1}^{k} X^{n_i}$$

define an endofunctor Σ^* on $\mathbf{Set}^*$: $\Sigma^*\,\langle X, \tau\rangle = \langle \Sigma X, \Sigma_X \tau\rangle$, where

$$\Sigma_X \tau = \mathrm{Cl}^{\cup}\Big(\bigoplus_{i=1}^{k} \tau^{\otimes n_i}\Big)$$

where $\mathrm{Cl}^{\cup}$ is the closure under arbitrary unions, and we write $\tau^{\otimes n_i}$ for $\underbrace{\tau \otimes \tau \otimes \cdots \otimes \tau}_{n_i \text{ times}}$. On arrows, given $f : \langle X, \tau\rangle \to \langle X', \tau'\rangle$, we define simply $\Sigma^* f = \Sigma f$. Preservation of identities and composition follows from the

fact that Σ is a functor. It suffices to show that Σf is a morphism in $\mathbf{Set}^*$, and indeed

$$\begin{aligned}
(\Sigma f)^* \Sigma_{X'} \tau' &= (\Sigma f)^* \operatorname{Cl}^{\cup}\Big(\bigoplus_{i=1}^{k} \tau'^{\otimes n_i}\Big) \\
&= \operatorname{Cl}^{\cup}(\Sigma f)^*\Big(\bigoplus_{i=1}^{k} \tau'^{\otimes n_i}\Big) \\
&= \operatorname{Cl}^{\cup}\Big(\bigoplus_{i=1}^{k} (f^*\tau')^{\otimes n_i}\Big) \\
&\subseteq \operatorname{Cl}^{\cup}\Big(\bigoplus_{i=1}^{k} \tau^{\otimes n_i}\Big) \qquad \text{(since } f^*\tau' \subseteq \tau\text{)} \\
&= \Sigma_X \tau
\end{aligned}$$

Theorem 20. Suppose that an endofunctor F lifts to a endofunctor F^*, and has an initial algebra $\psi : FN \to N$. Then $\psi : \langle FN, F_N P\rangle \to \langle N, P\rangle$ is the initial F^* algebra where P is the greatest fixpoint of the operator $\Psi(\tau) = (\psi^{-1})^* F_N \tau$.

Proof. First, by definition $\psi^* P = \psi^*(\psi^{-1})^* F_N P = F_N P$ and therefore ψ is a morphism in the total category.

Suppose that $h : \langle FX, F_X\tau\rangle \to \langle X, \tau\rangle$ is any F^*-algebra. Then $h^*\tau \subseteq F_X\tau$ and $h : FX \to X$ is an F-algebra, and we have a unique $k : N \to X$ such that $h \circ Fk = k \circ \psi$:

$$\begin{array}{ccc}
FN & \xrightarrow{Fk} & FX \\
{\scriptstyle\psi}\downarrow & & \downarrow{\scriptstyle h} \\
N & \xrightarrow[k]{} & X
\end{array}$$

It remains to verify that k is a morphism in the total category, i.e., that $k^*\tau \subseteq P$. This is done by proving that $k^*\tau$ is a postfixpoint of Ψ:

$$\begin{aligned}
\Psi(k^*\tau) &= (\psi^{-1})^* F_N (k^*\tau) \\
&\supseteq (\psi^{-1})^* (Fk)^* (F_X(\tau)) \qquad \text{(functoriality of } F\text{)} \\
&\supseteq (\psi^{-1})^* (Fk)^* h^* \tau \\
&= k^*\tau
\end{aligned}$$

P, as the greatest fixpoint of Ψ, is also the greatest postfixpoint of Ψ, hence $k^*\tau \subseteq P$. □

Corollary 21. For any syntactic endofunctor Σ, the functor Σ^* freely generates a monad T^* that lifts the monad T freely generated by Σ.

Proof. By definition, $T^* \langle X, \tau \rangle$ is the carrier of the initial $(\langle X, \tau \rangle + \Sigma^* -)$-algebra. Since Σ^* lifts Σ, by Theorem 20 we know that this initial algebra exists, and it is of the form $T^* \langle X, \tau \rangle = \langle TX, T_X\tau \rangle$, where $\psi : X + \Sigma TX \to TX$ is the initial $(X + \Sigma -)$-algebra, and $T_X\tau$ is the greatest fix point of the operator

$$\Psi\tau' = (\psi^{-1})^*(\tau \oplus \Sigma_{TX}\tau')$$

□

A similar corollary about a behaviour B^W cofreely generating a comonad D^W can be drawn from Theorem 14. These two corollaries allow us to apply Theorem 10 for the category $\mathbf{Set}^*$ and endofunctors Σ^* and B^W.

The following theorem is a crucial property of the endofunctor Σ^*. Indeed, varying the definition of Σ^* in our framework would lead to definition of various precongruence formats, but only as long as the following property holds.

Theorem 22. For any Σ^*-algebra $h : \langle \Sigma X, \Sigma_X\tau \rangle \to \langle X, \tau \rangle$, the specialisation preorder $\leq_\tau$ is a precongruence on $h : \Sigma X \to X$.

Proof. Take $\mathtt{f} : X^n \to \Sigma X$ to be one of the coproduct injections to ΣX, and consider elements $x_1, y_1, \ldots, x_n, y_n$ of X such that $x_i \leq_\tau y_i$ for $i = 1..n$. This means that

$$\langle x_1, \ldots, x_n \rangle \leq_{\tau^{\otimes n}} \langle y_1, \ldots, y_n \rangle$$

Observe that the closure operator $\mathrm{Cl}^\cup$ does not change specialisation preorders: for any test suite τ, we have $\leq_{\mathrm{Cl}^\cup \tau} = \leq_\tau$. Hence

$$\mathtt{f}\,\langle x_1, \ldots, x_n \rangle \leq_{\Sigma_X\tau} \mathtt{f}\,\langle y_1, \ldots, y_n \rangle$$

Now since h is a morphism in the total category, we know that $h^*\tau \subseteq \Sigma_X\tau$, hence

$$\mathtt{f}\,\langle x_1, \ldots, x_n \rangle \leq_{h^*\tau} \mathtt{f}\,\langle y_1, \ldots, y_n \rangle$$

and

$$h(\mathtt{f}\,\langle x_1,\ldots,x_n\rangle) \leq_\tau h(\mathtt{f}\,\langle y_1,\ldots,y_n\rangle)$$

□

We now have the technology needed to prove the main result of this section.

Consider a natural transformation $\lambda : \Sigma(\mathrm{id}\times B) \to BT$. By Theorem 10, the coalgebraic part of the initial λ-model has $N = T\emptyset$ as its carrier, and it is the intended operational model of λ. If $B = \mathcal{P}_\mathrm{f}(A \times -)$, then the intended operational model is the LTS generated by GSOS rules associated to λ. Let $k : N \to S$ be the final coalgebra morphism from the intended operational model to the final B-coalgebra. Assume B lifts to some B^* as before, and let $\langle S, M\rangle$ be the carrier of the initial B^*-coalgebra.

Theorem 23. If λ lifts to a natural transformation in the total category:

$$\lambda : \Sigma^*(\mathrm{id}\times B^*) \to B^*T^*.$$

then the specialisation preorder on k^*M is a precongruence on N.

Proof. By Theorem 10, there exists an initial λ-model (in $\mathbf{Set}^*$)

$$\Sigma^*\,\langle N,P\rangle \xrightarrow{\psi} \langle N,P\rangle \xrightarrow{\epsilon} B^*\,\langle N,P\rangle$$

and a final λ-model (in $\mathbf{Set}^*$)

$$\Sigma^*\,\langle S,M\rangle \xrightarrow{\delta} \langle S,M\rangle \xrightarrow{\varphi} B^*\,\langle S,M\rangle$$

where $\psi : \Sigma^*\,\langle N,P\rangle \to \langle N,P\rangle$ and $\varphi : \langle S,M\rangle \to B^*\,\langle S,M\rangle$ are respectively the initial Σ^*-algebra and the final B^*-coalgebra. By initiality (or finality) there exists a unique morphism $k : \langle N,P\rangle \to \langle S,M\rangle$ of λ-models so that *(i)* below is commutative.

$$\begin{array}{ccc}
\Sigma^*\,\langle N,P\rangle & \xrightarrow{\Sigma^*k} & \Sigma^*\,\langle S,M\rangle \\
\downarrow{\scriptstyle\psi} & & \downarrow{\scriptstyle\delta} \\
\langle N,P\rangle & \xrightarrow{k} & \langle S,M\rangle \\
\downarrow{\scriptstyle\epsilon} & & \downarrow{\scriptstyle\varphi} \\
B^*\,\langle N,P\rangle & \xrightarrow[B^*k]{} & B^*\,\langle S,M\rangle
\end{array}
\qquad
\begin{array}{ccc}
\Sigma^*\,\langle N,k^*M\rangle & \xrightarrow{\Sigma^*k} & \Sigma^*\,\langle S,M\rangle \\
\downarrow{\scriptstyle\psi} & & \downarrow{\scriptstyle\delta} \\
\langle N,k^*M\rangle & \xrightarrow{k} & \langle S,M\rangle \\
\downarrow{\scriptstyle\epsilon} & & \downarrow{\scriptstyle\varphi} \\
B^*\,\langle N,k^*M\rangle & \xrightarrow[B^*k]{} & B^*\,\langle S,M\rangle
\end{array}$$

(i) $\qquad$ (ii)

Our goal is to show that *(ii)* above a diagram in $\mathbf{Set}^*$. If all the morphisms are defined then its commutativity follows from the commutativity of *(i)*. By Theorem 15, $\epsilon : \langle N, k^*M\rangle \to B^* \langle N, k^*M\rangle$ is a B^*-coalgebra.

Now

$$\begin{aligned}\psi^* k^* M &= (\Sigma^* k)^* \delta^* M \\ &= (\Sigma k)^* \delta^* M \\ &\subseteq (\Sigma k)^* (\Sigma_S M) \qquad (\text{since } \delta \text{ is in } \mathbf{Set}^*) \\ &\subseteq \Sigma_X (k^* M) \qquad (\text{since } \Sigma^* \text{ is a functor})\end{aligned}$$

Thus $\psi : \langle N, k^*M\rangle \to B^* \langle N, k^*M\rangle$ is a Σ^*-algebra and by Theorem 22 the specialisation preorder of k^*M is a precongruence. □

7 Precongruence Formats for (almost) Free

In this section we consider a syntactic endofunctor Σ with a freely generated monad T, the behaviour functor $BX = \mathcal{P}_{\mathrm{f}}(A \times X)$, and a set R of GSOS rules with the corresponding natural transformation

$$\lambda : \Sigma(\mathrm{id} \times B) \to BT$$

The purpose is to describe syntactic conditions on R that would ensure that λ lifts to a natural transformation

$$\lambda : \Sigma^*(\mathrm{id} \times B^W) \to B^W T^*$$

where $W \in \{\mathsf{Tr}, \mathsf{CTr}, \mathsf{Fl}\}$. As a consequence of Theorem 23, such syntactic conditions ensure that the respective operational preorders are precongruences.

Theorem 24. If R is in Tr-format (Format 2), then

$$\lambda : \Sigma^*(Id \times B^{\mathsf{Tr}}) \to B^{\mathsf{Tr}} T^*$$

is a natural transformation in $\mathbf{Set}^*$.

Proof. It is enough to ensure that given an object $\langle X, \tau\rangle$ in $\mathbf{Set}^*$,

$$\lambda : \Sigma^*(\langle X, \tau\rangle \times B^{\mathsf{Tr}} \langle X, \tau\rangle) \to B^{\mathsf{Tr}} T^* \langle X, \tau\rangle$$

is a morphism in $\mathbf{Set}^*$; in other words, that for every test $V \in B^{\mathsf{Tr}}_{TX} T_X \tau$, $\lambda^{-1} V$ is a test in $\Sigma_{X \times BX}(\tau \bowtie B^{\mathsf{Tr}}_X \tau)$.

To do this, it will be useful to understand the nature of tests in $\Sigma_X \tau$, $T_X \tau$ and $B^{\mathsf{Tr}}_{TX} T_X \tau$, for a given $\langle X, \tau \rangle$.

Definition 25. For a syntactic Σ, a set X and a test suite τ on X, a *basic flat τ-check* on ΣX is a term

$$\gamma \in \Sigma \tau$$

A term $t \in \Sigma X$ *passes* a basic flat τ-check γ, if t can be obtained from γ by replacing every $V \in \tau$ by some $x \in V$.

The set of terms passing a given basic flat τ-check γ is denoted $\upsilon(\gamma)$. Sets of this kind will be called *basic flat τ-tests* on ΣX.

Lemma 26. For any Σ, $\langle X, \tau \rangle$, tests from $\Sigma_X \tau$ are exactly all unions of basic flat τ-tests on ΣX.

Proof. Take any test

$$V \in \bigoplus_{i=1}^{k} \tau^{\otimes n_i}$$

By definition, $V = V_1 + \cdots + V_k$, where $V_i \in \tau^{\otimes n_i}$. Then for every $1 \leq i \leq k$,

$$V_i = U_1 \times \cdots \times U_{n_i} \text{ where } U_j \in \tau$$

(in the above, $+$ and $\times$ denote disjoint union and cartesian product on sets). Note that $\gamma = \iota_i \langle U_1, \ldots, U_{n_i} \rangle$ is a basic flat τ-check, and

$$\iota_i(V_i) = \upsilon(\gamma)$$

where $\iota_i : X^{n_i} \to \Sigma X$ is the i-th coproduct injection into ΣX. Finally

$$V = V_1 + V_2 \cdots + V_k = \iota_1(V_1) \cup \iota_2(V_2) \cup \cdots \cup \iota_k(V_k)$$

which completes the presentation of V as a union of basic flat τ-tests. □

Definition 27. For a set X and a test suite τ on X, a *basic term τ-check* is a term

$$c \in T \tau$$

A term $t \in TX$ *passes* a basic term τ-check c, if t can be obtained from c by replacing every $V \in \tau$ by some $x \in V$.

The set of terms passing a given basic term τ-check c is denoted $\upsilon(c)$. Sets of this kind will be called *basic term τ-tests* on TX.

Lemma 28. For any Σ, $\langle X, \tau \rangle$, every test in $T_X\tau$ is a union of basic term τ-tests on TX.

Proof. Recall from Corollary ?? that $T_X\tau$ is the greatest fix point of the operator Ψ:

$$\Psi\tau' = (\psi^{-1})^*(\tau \oplus \Sigma_{TX}\tau')$$

The proof proceeds by constructing for all n and for any test $V \in \Psi^n(\mathcal{P}TX)$, a family $\Gamma_n(V)$ of basic term τ-checks from $T_n\tau$ such that

$$V \cap T_nX = \bigcup_{\gamma \in \Gamma_n(V)} \upsilon(\gamma)$$

where T_nX denotes the set of Σ-terms of depth at most n.

Given this construction, for any test $V \in T_X\tau$ we have

$$V = \bigcup_{n \in N} \bigcup_{\gamma \in \Gamma_n(V)} \upsilon(\gamma)$$

which will complete the proof.

The families $\Gamma_n(V)$ are constructed by induction on n. For the base case, take $\Gamma_0(V) = \emptyset$ for any V. Indeed,

$$V \cap T_0X = V \cap \emptyset = \bigcup_{\gamma \in \emptyset} \upsilon(\gamma)$$

For the induction step, take a test $V \in \Psi^n(\mathcal{P}TX)$. By definition, $V = V' + V''$ ($+$ denotes disjoint union of sets), where $V' \in \tau$, and $V'' \in \Sigma_{TX}\Psi^{n-1}(\mathcal{P}TX)$. By Lemma 26, for some indexing set I

$$V'' = \bigcup_{i \in I} \upsilon(\delta_i)$$

where each δ_i is a basic flat $\Psi^{n-1}(\mathcal{P}TX)$-check on ΣTX, i.e.

$$\delta_i \in \Sigma(\Psi^{n-1}(\mathcal{P}TX))$$

By the inductive assumption, for every test $U \in \Psi^{n-1}(\mathcal{P}TX)$ there exist a family $\Gamma_{n-1}(U)$ of basic term τ-checks such that

$$U \cap T_{n-1}X = \bigcup_{u \in \Gamma_{n-1}(U)} \upsilon(u)$$

Now for any δ_i as above, take $\delta_i' = \delta_i[\sigma]$, where

$$\sigma(U) = \bigcup_{u \in \Gamma_{n-1}(U)} \upsilon(u)$$

is a substitution from $\Psi^{n-1}(\mathcal{P}TX)$ to $\mathcal{P}TX$. Obviously, each δ_i' is a basic flat $(\mathcal{P}TX)$-check and

$$\begin{aligned} \upsilon(\delta_i') &= \upsilon(\delta_i) \cap \Sigma T_{n-1} X \\ \iota_2(\upsilon(\delta_i')) &= \iota_2(\upsilon(\delta_i)) \cap T_n X \end{aligned}$$

where $\iota_2 : \Sigma TX \to TX$ is the coproduct inclusion.

Moreover, for any δ_i as above, consider the family of basic term τ-checks:

$$\Theta(\delta_i) = \{\, \iota_2(\delta_i[\sigma']) \,:\, \sigma'(U) \in \Gamma_{n-1}(U) \,\} \subseteq T_n \tau$$

where $\iota_2 : \Sigma T_{n-1}\tau \to T_n\tau$ is the coproduct inclusion, and σ' ranges on substitutions from $\Psi^{n-1}(\mathcal{P}TX)$ to $T_{n-1}\tau$.

From definition of υ, it is easy to see that

$$\bigcup_{\gamma \in \Theta(\delta_i)} \upsilon(\gamma) = \iota_2(\upsilon(\delta_i'))$$

Now take

$$\Gamma_n(V) = \{\iota_1(V')\} \cup \bigcup_{i \in I} \Theta(\gamma_i)$$

and check that

$$\begin{aligned} V \cap T_n X &= (V' + V'') \cap T_n X \\ &= (\iota_1(V') \cup \iota_2(V'')) \cap T_n X \\ &= (\iota_1(V') \cap T_n X) \cup (\iota_2(V'') \cap T_n X) \\ &= \iota_1(V') \cup \Big(\bigcup_{i \in I} \iota_2(\upsilon(\delta_i)) \cap T_n X \Big) \\ &= \upsilon(\iota_1(V')) \cup \Big(\bigcup_{i \in I} \iota_2(\upsilon(\gamma_i')) \Big) \\ &= \upsilon(\iota_1(V')) \cup \Big(\bigcup_{i \in I} \bigcup_{\gamma \in \Theta(\delta_i)} \upsilon(\gamma) \Big) \end{aligned}$$

This completes the induction step, and the proof of Lemma 28. □

Definition 29. For a given $\langle X, \tau \rangle$, a *basic positive τ-check* on BX is an expression of the form $\xrightarrow{a} V$, where $a \in A$ and $V \in \tau$. A set $\beta \in BX$ *passes* such a check, if there is some $\langle a, x \rangle \in \beta$ such that $x \in V$. As usual, the test on BX associated to a basic positive τ-check c is denoted $\upsilon(c)$ and is called a *basic positive τ-test*.

Similarly, for a given $\langle X, \tau \rangle$, a *basic positive term τ-check* on BTX is an expression of the form $\xrightarrow{a} \gamma$, where $a \in A$ and γ is a basic term τ-test on TX. The definition of passing and of the *basic positive term τ-test* $\upsilon(\xrightarrow{a} \gamma)$ is as usual.

Lemma 30. A test in $B^{\mathsf{Tr}}_{TX} T_X \tau$ is either the always true test BTX, or a union of basic positive term τ-tests.

Proof. If a test $V \in B^{\mathsf{Tr}}_{TX} T_X \tau$ is not equal to BTX, then

$$V = w_{\langle a \rangle} \circ BV' = \{\, \beta \in BTX \,:\, \langle a, t \rangle \in \beta, t \in V' \,\}$$

for some $V' \in T_X \tau$. But then, by Lemma 28, V' is a union of basic term τ-tests:

$$V' = \bigcup_{i \in I} \upsilon(\gamma_i)$$

hence

$$V = \bigcup_{i \in I} \{\, \beta \in BTX \,:\, \langle a, t \rangle \in \beta, t \in \upsilon(\gamma_i) \,\} = \bigcup_{i \in I} \upsilon(\xrightarrow{a} \gamma_i)$$

□

We shall now prove Theorem 24. By Lemma 30, it is enough to show that for any $\langle X, \tau \rangle$, and for any given basic positive term τ-check $\xrightarrow{a} \gamma$, there exists a family of basic flat $(\tau \bowtie B^{\mathsf{Tr}}_X \tau)$-checks $\delta_1, \delta_2, \ldots, \delta_m$ such that

$$\lambda^{-1} \upsilon(\xrightarrow{a} \gamma) = \bigcup_{i=1}^{m} \upsilon(\delta_i)$$

To achieve this, fix a basic positive term τ-check $\xrightarrow{a} \gamma$, and take any rule $\rho \in R$, which has the conclusion of the form

$$\mathtt{f}(\mathtt{x}_1, \ldots \mathtt{x}_n) \xrightarrow{a} \mathtt{t}$$

for some construct $\mathtt{f}$ with arity n, and such that $\mathtt{t}$ can be obtained from γ by replacing each $V \in \tau$ with some $\mathtt{x} \in \Xi$. Since no variable occurs in

$\mathtt{t}$ more than once, this gives a function $\sigma : \Xi \to \tau$ such that $\gamma = \mathtt{t}[\sigma]$. Without loss of generality, assume that $\sigma(\mathtt{x}) = X$ if $\mathtt{x}$ does not occur in $\mathtt{t}$.

Now for each $1 \leq i \leq n$ take a test $V_i \in \tau \bowtie B_X\tau$ as follows:

- If $\mathtt{x}_i$ does not occur in any premise of ρ, then take $V_i = \sigma(\mathtt{x}_i) \times BX$.
- If $\mathtt{x}_i$ occurs in a premise $\mathtt{x}_i \xrightarrow{b_i} \mathtt{y}_i$, then take $V_i = X \times \upsilon(\xrightarrow{b_i} \sigma(\mathtt{y}_i))$

Note that the syntactic restrictions of the Tr-format ensure that the above definition is complete and unambiguous.

Having defined the tests V_i, consider a basic flat $\tau \bowtie B_X\tau$-check

$$\delta_\rho = \mathtt{f}\,\langle V_1, \ldots, V_n\rangle$$

We will show that $\upsilon(\delta_\rho) = \rho_X^{-1}\upsilon(\xrightarrow{a} \gamma)$. To this end, take any

$$r = \mathtt{f}\,\langle\langle x_1, \beta_1\rangle, \ldots, \langle x_n, \beta_n\rangle\rangle \in \Sigma(X \times BX)$$

and check that the following are equivalent:

r passes δ_ρ

$\iff$ $\langle x_i, \beta_i\rangle \in V_i$ for all $1 \leq i \leq n$

$\iff$ For each $1 \leq i \leq n$, either $\mathtt{x}_i$ does not occur in any premise and $x_i \in \sigma(\mathtt{x}_i)$, or $\mathtt{x}_i$ occurs in a premise $\mathtt{x}_i \xrightarrow{b_i} \mathtt{y}_i$ and $\langle b_i, y_i\rangle \in \beta_i$ for some $y_i \in \sigma(\mathtt{y}_i)$

$\iff$ θ mapping each $\mathtt{x}_i$ to x_i, and $\mathtt{y}_i$ to y_i satisfies the conditions described in the proof of Theorem 9. Moreover, $\theta(\mathtt{x}) \in \sigma(\mathtt{x})$ for all $\mathtt{x} \in \Xi$.

$\iff$ $\langle a, \mathtt{t}[\theta]\rangle \in \rho_X r$, and $\mathtt{t}[\theta]$ passes γ.

$\iff$ $\rho_X r$ passes $\xrightarrow{a} \gamma$.

This shows that $\upsilon(\delta_\rho) = \rho_X^{-1}\upsilon(\xrightarrow{a} \gamma)$.

Now by definition of λ from the family of ρ_X for $\rho \in R$,

$$\lambda^{-1}(\upsilon(\xrightarrow{a} \gamma)) = \bigcup_\rho \upsilon(\rho_X)$$

where the summation is made over all rules ρ with the conclusion as described above.

This completes the proof of Theorem 24. $\square$

Theorem 31. If R is in CTr-format, then

$$\lambda : \Sigma^*(Id \times B^{\mathsf{CTr}}) \to B^{\mathsf{CTr}}T^*$$

is a natural transformation in $\mathbf{Set}^*$.

Proof. As before, we show that for any $\langle X, \tau \rangle$, for every test $V \in B^{\mathsf{CTr}}_{TX} T_X$ there is a test $V' \in \Sigma_X(\tau \bowtie B^{\mathsf{CTr}}_X \tau)$ such that $\lambda^{-1}V = V'$.

Definition 32. For a set of labels $Q \subseteq A$, the *Q-failure check* is denoted $\stackrel{Q}{\not\longrightarrow}$. Instead of $\stackrel{A}{\not\longrightarrow}$, we will sometimes write $\not\longrightarrow$. For any set X, a set $\beta \in BX$ *passes* the Q-failure check if

$$\beta \cap \{ \langle a, t \rangle \ : \ a \in Q \} = \emptyset$$

For any X, the test on BX associated to the Q-failure check will be denoted $\upsilon(\stackrel{Q}{\not\longrightarrow})$, and will be called the *Q-failure test*.

Lemma 33. A test $V \in B^{\mathsf{CTr}}_{TX} T_X$ is either the always true test BTX, or the A-failure test $\upsilon(\not\longrightarrow)$, or a union of basic positive term τ-tests.

Proof. If a test $V \in B^{\mathsf{CTr}}_{TX} T_X \tau$ is not equal to BTX, then either $V = w_{\langle a \rangle} \circ BV'$ for some $V' \in T_X \tau$, or $V = w_{rA} \circ BV'$ for some $V' \in T_X \tau$. In the former case, V is a union of basic positive term τ-tests, as shown in Lemma 30. In the latter case,

$$w_{rA} \circ BV' = \{\emptyset\} = \upsilon(\not\longrightarrow)$$

$\square$

To prove Theorem 31, it is enough to consider single basic positive term τ-tests on BTX, and the A-failure test on BTX. For the positive tests, we proceed as in the proof of Theorem 24, except that when constructing the tests V_i we consider one additional case:

- If $\mathtt{x}_i$ occurs in ρ in a negative premise $\mathtt{x}_i \stackrel{a}{\not\longrightarrow}$ (and hence, it occurs in a premise $\mathtt{x}_i \stackrel{b}{\not\longrightarrow}$ for any $b \in A$), then take $V_i = X \times \upsilon(\not\longrightarrow)$.

The syntactic restrictions of CTr-format ensure that the definition of V_i's extended this way is complete and unambiguous.

The rest of the argument remains as in the case of Tr-format.

For the A-failure test $\upsilon(\not\longrightarrow)$ on BTX, for any language construct $\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n)$ take a sequence of CTr-testing sets $P_1, \ldots, P_{k_\mathtt{m}}$ that satisfy the condition described in the definition of CTr-format. For each P_i and each variable $\mathtt{x}_j$, define a test $V_{ij} \in \tau \bowtie B_X^{\mathsf{CTr}}\tau$ as follows:

- If, for some $a \in A$, both $\mathtt{x}_j \xrightarrow{a}$ and $\mathtt{x}_j \not\xrightarrow{a}$ belong to P_i, then take $V_{ij} = \emptyset$.
- If, for all $a \in A$, $\mathtt{x}_j \xrightarrow{a} \in P_i$, and if for all $b \in A$, $\mathtt{x}_j \not\xrightarrow{b} \notin P_i$, then take $V_{ij} = X \times \upsilon(\not\longrightarrow)$.
- If, for all $a \in A$, $\mathtt{x}_j \xrightarrow{a} \notin P_i$, and if for some $b \in A$, $\mathtt{x}_j \not\xrightarrow{b} \in P_i$, then take $V_{ij} = X \times \upsilon(\xrightarrow{b} X)$.

Note that the definition of a CTr-testing set ensures that the above definition is complete and unambiguous.

Now for any P_i consider the basic flat $\tau \bowtie B_X^{\mathsf{CTr}}\tau$-check

$$\delta_{\mathtt{f}i} = \mathtt{f}\,\langle V_{i1}, \ldots, V_{in}\rangle$$

Recall the definition of the function f_X in the proof of Theorem 9. We will now show that

$$\bigcup_{i=1}^{k_\mathtt{f}} \upsilon(d_{\mathtt{f}i}) = f_X^{-1}\upsilon(\not\longrightarrow)$$

To this end, consider any $r \in \Sigma(X \times BS)$ of the form

$$r = \iota_m\,\langle\langle x_1, \beta_1\rangle, \ldots, \langle x_n, \beta_n\rangle\rangle$$

and assume that $f_X r$ does *not* pass $\not\longrightarrow$, i.e., that $f_X r \neq \emptyset$. This means that there exists a rule $\rho \in R$ with source $\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n)$, and a substitution $\sigma : \Xi \to X$ such that

$$\begin{array}{l}\sigma\mathtt{x}_i = x_i\\ \forall i \leq n \forall j \leq m_i.\, \langle a_{ij}, \sigma(\mathtt{y}_{ij})\rangle \in \beta_i\\ \forall i \leq n \forall j \leq n_i \forall x \in X.\, \langle b_{ij}, x\rangle \notin \beta_i\end{array}$$

where $m_i, n_i, a_{ij}, b_{ij}, \mathtt{y}_{ij}$ are taken from ρ.

By the first part of condition 2 of CTr-format, there exists a sequence $p_1, \ldots, p_{k_\mathtt{f}}$ such that $p_i \in P_i$ and each p_i is completed by some premise of ρ. Take any $1 \leq i \leq k_\mathtt{f}$. In ρ there is a premise that completes p_i. If p_i is a positive semiliteral $\mathtt{x}_j \xrightarrow{c}$ (for some $\mathtt{x}_j$, c), then $\beta_j \neq \emptyset$, hence $\langle x_j, \beta_j \rangle \notin V_{ij}$ and r does not pass $d_{\mathtt{f}i}$. Similarly, if p_i is a negative semiliteral $\mathtt{x_j} \not\xrightarrow{c}$, then $\beta_j = \emptyset$ and again $\langle x_j, \beta_j \rangle \notin V_{ij}$ and r does not pass $d_{\mathtt{f}i}$.

Conversely, assume that r does *not* pass $d_{\mathtt{f}i}$ for any $1 \leq i \leq k_\mathtt{f}$. This means that for each i, there is a j such that $\langle x_j, \beta_j \rangle \notin V_{ij}$. For each $1 \leq i \leq k_\mathtt{f}$, take p_i to be any element of P_i that has the respective $\mathtt{x}_j$ on the left-hand side. By the second part of condition 2 of CTr-format, there exists a rule $\rho \in R$ with source $\mathtt{f}(\mathtt{x}_1, \ldots, \mathtt{x}_n)$, such that each premise of ρ completes some p_i. This means that there exists a substitution $\sigma : \Xi \to X$ satisfying all the conditions mentioned above, hence $f_X r \neq \emptyset$ and $f_X r$ does not pass $\not\rightarrow$.

From this, by definition of λ as in the proof of Theorem 9,

$$\bigcup_{\mathtt{f} \in \Sigma} \bigcup_{i=1}^{k_\mathtt{f}} \upsilon(d_{\mathtt{f}i}) = \lambda^{-1} \upsilon(\not\rightarrow)$$

This completes the proof of Theorem 31. □

Theorem 34. If R is in Fl-format, then

$$\lambda : \Sigma^*(Id \times B^{\mathsf{Fl}}) \to B^{\mathsf{Fl}} T^*$$

is a natural transformation in $\mathbf{Set}^*$.

Proof. As before, we show that for any $\langle X, \tau \rangle$, for every test $V \in B^{\mathsf{Fl}}_{TX} T_X$ there is a test $V' \in \Sigma_X(\tau \bowtie B^{\mathsf{Fl}}_X \tau)$ such that $\lambda^{-1} V = V'$.

As before, we begin with a characterisation of tests in $B^{\mathsf{Fl}}_{TX} T_X$:

Lemma 35. A test $V \in B^{\mathsf{Fl}}_{TX} T_X$ is either the always true test BTX, or the Q-failure test $\upsilon(\not\rightarrow)$ for some $Q \subseteq A$, or a union of basic positive term τ-tests.

Proof. As in the proof of Lemma 33. □

The proof proceeds much the same as in Theorem 31. For the positive tests V, we proceed as in the proof of Theorem 24, except that when constructing the tests V_i we consider one additional case:

- If $\mathtt{x}_i$ occurs in ρ in some negative premises, then take $V_i = X \times \upsilon(\overset{Q_i}{\not\longrightarrow})$, where $Q_i = \{\, a \in A : \mathtt{x}_i \overset{a}{\not\longrightarrow} \text{ is a premise in } \rho \,\}$.

Again, the syntactic restrictions of FI-format ensure that the definition of V_i's extended this way is complete and unambiguous.

The rest of the argument remains as in the case of Tr-format.

For the Q-failure test $\upsilon(\overset{Q}{\not\longrightarrow})$ for some $Q \subseteq A$, we proceed as in the proof of Theorem 31, except that we construct the tests V_{ij} in a slightly different manner:

- If, for some $a \in A$, both $\mathtt{x}_j \xrightarrow{a}$ and $\mathtt{x}_j \overset{a}{\not\longrightarrow}$ belong to P_i, then take $V_{ij} = \emptyset$.
- If for some $a \in A$, $\mathtt{x}_j \xrightarrow{a} \in P_i$, and if for all $b \in A$, $\mathtt{x}_j \overset{b}{\not\longrightarrow} \notin P_i$, then take $V_{ij} = X \times \upsilon(\overset{Q_j}{\not\longrightarrow})$, where $Q_j = \{\, a \in A : \mathtt{x}_j \xrightarrow{a} \in P_i \,\}$.
- If, for all $a \in A$, $\mathtt{x}_j \xrightarrow{a} \notin P_i$, and if for some $b \in A$, $\mathtt{x}_j \overset{b}{\not\longrightarrow} \in P_i$, then take $V_{ij} = X \times \upsilon(\xrightarrow{b} X)$.

(note that the first and the third case are as in the case of CTr-format).

Note that the definition of a FI-testing set ensures that the above definition is complete and unambiguous.

The rest of the argument remains the same as in the case of CTr-format. □

8 Conclusions

We have presented an abstract coalgebraic approach to the description of various operational preorders, via a fibration of test suites. In Theorems 16, 17 and 18 we illustrated this approach on the trace preorder, the completed trace preorder and the failures preorder. Combined with bialgebraic methods, this framework allows the derivation of syntactic subformats of GSOS which guarantee that the aforementioned operational preorders are precongruences. Theorem 23 is a guideline in the search for such formats, and Theorems 24, 31 and 34 are applications of the framework.

The generality and abstractness of Theorem 23 prompted us to coin the expression 'precongruence format for free'. However, it must be stressed that to derive a syntactic format for a given operational preorder remains a non-trivial task. Indeed, the proofs of Theorems 24, 31 and 34 are quite long and technical. The expression 'for free' reflects the fact that Theorem 23 lets us prove precongruence properties without considering the global behaviour (e.g. traces) of processes. Instead, one considers only simple tests on processes, corresponding intuitively to single modalities.

Related abstract approaches to operational preorders and equivalences include those based on modal characterisations [10] and quantales [1]. In the latter framework, no syntactic issues have been addressed. In the former, some general precongruence formats have been obtained by attempting to decompose modal formulae according to given operational rules [7]. This technique bears some resemblance to our approach, and the precise connections have to be investigated.

There are several possible directions of future work. Firstly, the approach presented here can be extended to deal with other operational preorders and equivalences described in literature. Secondly, one can move from the GSOS format (and its subformats) to the more general (safe) ntree format [9], which can also be formalised in the bialgebraic framework [30]. Thirdly, the abstract framework of test suites seems to be general enough to cover other notions of process behaviour (e.g. involving store), or even other underlying categories (e.g. complete partial orders instead of sets). It may prove interesting to formalise various operational preorders in such cases and to find precongruence formats for them.

References

[1] S. Abramsky and S. Vickers. Quantales, observational logic and process semantics. *Math. Struct. in Comp. Sci.*, 3:161–227, 1993.

[2] L. Aceto, W. Fokkink, and C. Verhoef. Structural operational semantics. In J. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*. Elsevier, 1999.

[3] P. Aczel and N. Mendler. A final coalgebra theorem. In D. H. Pitt et al., editors, *Proc. CTCS'89*, volume 389 of *LNCS*, pages 357–365, 1989.

[4] M. Barr. Terminal coalgebras in well-founded set theory. *Theoretical Computer Science*, 114:299–315, 1993.

[5] B. Bloom. When is partial trace equivalence adequate? *Formal Aspects of Computing*, 6:25–68, 1994.

[6] B. Bloom, W. Fokkink, and R. J. van Glabbeek. Precongruence formats for decorated trace preorders. In *Logic in Computer Science*, pages 107–118, 2000.

[7] B. Bloom, W.J Fokkink, and R.J van Glabbeek. Precongruence formats for decorated trace semantics. *ACM Transactions on Computational Logic*. To appear.

[8] B. Bloom, S. Istrail, and A. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42:232–268, 1995.

[9] W. Fokkink and R. van Glabbeek. Ntyft/ntyxt rules reduce to ntree rules. *Information and Computation*, 126:1–10, 1996.

[10] R. J. van Glabbeek. The linear time-branching time spectrum I. In J. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*. Elsevier, 1999.

[11] J. F. Groote. Transition system specifications with negative premises. *Theoret. Comput. Sci.*, 118:263–299, 1993.

[12] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32:137–161, 1985.

[13] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[14] B. Jacobs. *Categorical Logic and Type Theory*, volume 141 of *Studies in Logic and the Foundations of Mathematics*. North Holland, Elsevier, 1999.

[15] A. Joyal, M. Nielsen, and G. Winskel. Bisimulation from open maps. *Information and Computation*, 127(2):164–185, 1996.

[16] S. Mac Lane. *Categories for the Working Matematician*. Springer-Verlag, second edition, 1998.

[17] J. Leifer and R. Milner. Deriving bisimulation congruences for reactive systems. In *Int. Conf. on Concurrency Theory, CONCUR 00*, Lecture Notes in Computer Science, pages 243–258. Springer Verlag, 2000.

[18] R. Milner and D. Sangiorgi. Barbed bisimulation. In *Proceedings of ICALP 92': Automata, Languages and Programming*, volume 623 of *Lecture Notes in Computer Science*. Springer-Verlag, 1992.

[19] D. M. Park. Concurrency on automata and infinite sequences. In P. Deussen, editor, *Conf. on Theoretical Computer Science*, volume 104 of *Lecture Notes in Computer Science*. Springer Verlag, 1981.

[20] G. Plotkin. Bialgebraic semantics (and recursion). Unpublished notes.

[21] G. Plotkin. A structural approach to operational semantics. DAIMI Report FN-19, Computer Science Department, Aarhus University, 1981.

[22] G. Plotkin. Bialgebraic semantics and recursion (extended abstract). In A. Corradini, M. Lenisa, and U. Montanari, editors, *Electronic Notes in Theoretical Computer Science*, volume 44. Elsevier Science Publishers, 2001.

[23] A. W. Roscoe. *The Theory and Practice of Concurrency*. Prentice Hall, 1997.

[24] J. Rutten and D. Turi. Initial algebra and final coalgebra semantics for concurrency. In J. de Bakker et al., editors, *Proc. of the REX workshop A Decade of Concurrency – Reflections and Perspectives*, volume 803 of *LNCS*, pages 530–582. Springer-Verlag, 1994.

[25] V. Sassone and P. Sobociński. Deriving bisimulation congruences: A 2-categorical approach. *Electronic Notes in Theoretical Computer Science*, 68(2), 2002.

[26] P. Sewell. From rewrite rules to bisimulation congruences. *Lecture Notes in Computer Science*, 1466:269–284, 1998.

[27] R. de Simone. Higher-level synchronising devices in Meije-SCCS. *Theoret. Comput. Sci.*, 37:245–267, 1985.

[28] D. Turi. Fibrations and bisimulation. Unpublished notes.

[29] D. Turi. *Functorial Operational Semantics and its Denotational Dual*. PhD thesis, Vrije Universiteit, Amsterdam, 1996.

[30] D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *Proceedings 12th Ann. IEEE Symp. on Logic in Computer Science, LICS'97, Warsaw, Poland, 29 June – 2 July 1997*, pages 280–291. IEEE Computer Society Press, 1997.

[31] Frits W. Vaandrager. On the relationship between process algebra and input/output automata. In *Logic in Computer Science*, pages 387–398, 1991.

Recent BRICS Report Series Publications

RS-03-18 **Bartek Klin and Paweł Sobociński.** ***Syntactic Formats for Free: An Abstract Approach to Process Equivalence.*** **April 2003. 41 pp.**

RS-03-17 **Luca Aceto, Jens Alsted Hansen, Anna Ingólfsdóttir, Jacob Johnsen, and John Knudsen.** ***The Complexity of Checking Consistency of Pedigree Information and Related Problems.*** **2003.**

RS-03-16 **Ivan B. Damgård and Mads J. Jurik.** ***A Length-Flexible Threshold Cryptosystem with Applications.*** **March 2003. 19 pp.**

RS-03-15 **Anna Ingólfsdóttir.** ***A Semantic Theory for Value–Passing Processes Based on the Late Approach.*** **March 2003. 49 pp.**

RS-03-14 **Mads Sig Ager, Dariusz Biernacki, Olivier Danvy, and Jan Midtgaard.** ***From Interpreter to Compiler and Virtual Machine: A Functional Derivation.*** **March 2003. 36 pp.**

RS-03-13 **Mads Sig Ager, Dariusz Biernacki, Olivier Danvy, and Jan Midtgaard.** ***A Functional Correspondence between Evaluators and Abstract Machines.*** **March 2003. 28 pp.**

RS-03-12 **Mircea-Dan Hernest and Ulrich Kohlenbach.** ***A Complexity Analysis of Functional Interpretations.*** **February 2003. 70 pp.**

RS-03-11 **Mads Sig Ager, Olivier Danvy, and Henning Korsholm Rohde.** ***Fast Partial Evaluation of Pattern Matching in Strings.*** **February 2003. 14 pp. To appear in Leuschel, editor,** ***ACM SIGPLAN Workshop on Partial Evaluation and Semantics-Based Program Manipulation,*** **PEPM '03 Proceedings, 2003.**

RS-03-10 **Federico Crazzolara and Giuseppe Milicia.** ***Wireless Authentication in χ-Spaces.*** **February 2003. 20 pp.**

RS-03-9 **Ivan B. Damgård and Gudmund Skovbjerg Frandsen.** ***An Extended Quadratic Frobenius Primality Test with Average and Worst Case Error Estimates.*** **February 2003. 53 pp.**

RS-03-8 **Ivan B. Damgård and Gudmund Skovbjerg Frandsen.** ***Efficient Algorithms for gcd and Cubic Residuosity in the Ring of Eisenstein Integers.*** **February 2003. 11 pp.**

RS-03-7 **Claus Brabrand, Michael I. Schwartzbach, and Mads Vanggaard.** ***The METAFRONT System: Extensible Parsing and Transformation.*** **February 2003. 24 pp.**