Familial monads and structural operational semantics

Tom Hirschowitz

Familial monads and structural operational semantics

TOM HIRSCHOWITZ, Univ. Grenoble Alpes, Univ. Savoie Mont Blanc, CNRS, LAMA, France

We propose an abstract framework for structural operational semantics, in which we prove that under suitable hypotheses bisimilarity is a congruence. We then refine the framework to prove soundness of bisimulation up to context, an efficient method for reducing the size of bisimulation relations. Finally, we demonstrate the flexibility of our approach by reproving known results about congruence of bisimilarity and soundness of bisimulation up to context, in three variants of the π-calculus.

1 INTRODUCTION

Motivation

Structural operational semantics [16] is a method for specifying the dynamics of programming languages by induction on their syntax. This means that one describes the behaviour of a program in terms of its components, a feature often called *compositionality*. An important issue in structural operational semantics is the extent to which compositionality entails good behaviour of the generated labelled transition system. In this paper, we consider two particular questions: *congruence of bisimilarity* and *soundness of bisimulation up to context*.

The former is a long-standing problem in structural operational semantics. Bisimilarity is probably the most widely used behavioural equivalence, and congruence of bisimilarity essentially amounts to substitutivity: given two bisimilar program fragments P and Q (which we denote by $P \sim Q$), do we have $C[P] \sim C[Q]$ for any context C? Bisimilarity is famously known not to be a congruence in general, e.g., in the π-calculus [21, §2.2.1].

Our second object of study is bisimulation up to context [17, 21], an efficient variant of bisimulation, which often produces the same results using simpler relations. However, it is sometimes unsound, in the sense that bisimilarity up to context may not entail bisimilarity. Just like congruence of bisimilarity, soundness of bisimulation up to context has proved to be a subtle matter.

The difficulty of these questions, particularly the former, led to a rich variety of syntactic *formats* [15], which ensure good behaviour of the generated labelled transition system, up to some constraints on the considered specification. Despite their diversity, formats have a lot in common, both in definitions and in proof schemes.

This commonality motivated *functorial operational semantics* [11, 24], a unifying theory of formats, in which specifications are recast as distributive laws of a comonad over a monad. The approach has been deeply developed in the set-based case, particularly for congruence of bisimilarity but also for soundness of bisimulation up to context (among others) [1]. However, the picture gets more complex in the presence of variable binding, as, e.g., in the impressive work by Fiore and Staton [7]. A simpler framework is considered in Staton [23], which makes sense in any $\prod W$-pretopos – typically presheaf toposes. Although the format is fairly involved, a great benefit is that variable binding is treated almost transparently.

However, despite its generality and power, the work of Fiore and Staton [7] and Staton [23] has not caught on as widely as one might have hoped among more practical operational semanticists. A reason for this might be that we should distinguish between two different goals, both of which are crucial for a theory of structural operational semantics to be adopted by the community: (1) the first goal is a high-level abstract language for *reasoning* on structural operational semantics,

Author's address: Tom Hirschowitz, Univ. Grenoble Alpes, Univ. Savoie Mont Blanc, CNRS, LAMA, 73000, Chambéry, France, tom.hirschowitz@univ-smb.fr.

as close as possible to concrete intuitions, (2) the second is a generic toolbox for *producing* well-behaved structural operational semantics from more basic data. It seems fair to say that, until now, most work on the first goal was done with a view to the second one, i.e., with formats in mind. In this paper, instead, our aim is to find the right level of generality, i.e., the right language, for standard proof schemes for congruence of bisimilarity and soundness of bisimulation up to context to apply.

Contributions

Our first contribution is the introduction of *transition categories*, which are categories equipped with a selection of cospans, thought of as the set of transition labels. Transition categories support a notion of bisimulation defined by lifting, much as in Joyal et al. [10]. A structural operational semantics specification is then a monad $\mathscr{T}$ on the considered transition category, which embodies the syntax and derivation rules for transitions. Finally, models of $\mathscr{T}$ are simply $\mathscr{T}$-algebras.

Our first main result (Corollary 3.26) states that whenever $\mathscr{T}$ satisfies a certain *familiality* property [4, 6, 8, 25] and the considered $\mathscr{T}$-algebras, say X and Y, are *compositional*, in the sense that both structure maps $\mathscr{T}(X) \to X$ and $\mathscr{T}(Y) \to Y$ are functional bisimulations, then bisimilarity (between states of X and Y) is a congruence.

We then turn to soundness of bisimulation up to context in §4. The definition by lifting is too rigid to directly accomodate bisimulation up to context, so we introduce a notion of *pre-bisimulation*. Under a mild additional hypothesis, using an appropriate weak factorisation system, we show that any pre-bisimulation embeds into some bisimulation. We then define *pre-bisimulation up to context*, which agrees with standard bisimulation up to context in examples. Our second main result (Corollary 4.13) is that, up to a refinement of the familiality hypothesis, any pre-bisimulation up to context embeds into some bisimulation, i.e., pre-bisimulation up to context is sound. This is to our knowledge the first general soundness result for bisimulation up to context covering calculi with variable binding.

Finally, we demonstrate the flexibility of our framework in §5 by analysing the failure of bisimilarity to be a congruence in the π-calculus. We first pinpoint where the hypotheses of Corollary 3.26 fail, namely: the structure map $\mathscr{T}(Pi) \to Pi$ is not a functional bisimulation, which we interpret as the input axiom being non-compositional. We then recast in our setting two standard ways of working around this issue: (1) prove the weaker claim that bisimilarity is a *non-input congruence* [21]; (2) restrict attention to a more constrained notion, *wide-open* bisimulation [7, 19, 23], to which Corollary 3.26 applies.

Related work

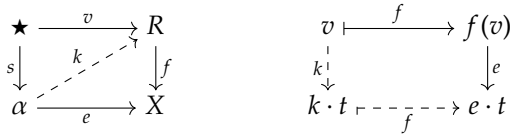
We only know of two other abstract accounts of structural operational semantics covering both syntax and models, and proving congruence of bisimilarity: functorial operational semantics and Staton [23]. The clearest novelty of our approach compared to them is that, to our knowledge, they do not cover soundness of bisimulation up to context in the presence of binding. Another distinctive feature is the crucial role of familiality. In particular, Staton [23, Theorem 12] is very close in spirit to our Corollary 3.26, but beyond the fact that it lives in a different setting, it only assumes that the considered monad preserves functional bisimulations, while familiality allows us to prove it. Less closely related work includes presheaf models and their generalisations [5, 10], which emphasise semantical, rather than operational aspects. Furthermore, their representation of labelled transition systems as presheaves markedly differs from ours, in that, e.g., any finite but cyclic labelled transition system may be finitely represented in our approach, while it has to be represented by an infinite presheaf in theirs. Our approach is also more economical for defining

bisimulation, by only lifting against one morphism per label, instead of one morphism per trace extension. Finally, our $\mathbf{T}_s$-familial monads are a specialisation of *cellular* analytic functors [8].

Overview

Before delving into details, let us give a non-technical overview. A transition category is much like a category of labelled transition systems (over a fixed set of labels), with functional simulations as morphisms. Sticking to the untyped case for simplicity, there is thus an object, say $\star$, consisting of just one state. Similarly, for each label α, there is an object consisting of just one α-transition between two states. Thus, giving a morphism $\star \to X$ to some object X is equivalent to choosing a state in X, and likewise giving a morphism $e\colon \alpha \to X$ is equivalent to choosing an α-transition in X. Furthermore, taking the source of e is just pre-composing with the source morphism $s\colon \star \to \alpha$. So the source of e is $e \circ s$. Symmetrically, its target is $\star \xrightarrow{t} \alpha \xrightarrow{e} X$.

Now, we may define bisimulation diagrammatically, as follows. A relation $R \hookrightarrow X \times Y$ is a bisimulation iff for all labels α and commuting squares as below left, there is a lifting k that makes



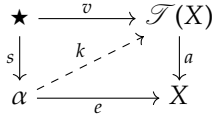
(1)

both triangles commute, where f denotes the composite $R \hookrightarrow X \times Y \xrightarrow{\pi} X$ (and symmetrically for Y). Indeed, as we saw, v denotes a state in R, i.e., a related pair (x, y), e an α-transition in X, and commutation of the square says that $f(v)$ is the source of e. We are thus in a situation like above right. Existence of k then says that there is an α-transition k in R with source v, mapped by f to e, as above right, just as in the standard definition of bisimulation.

The next step is to view structural operational semantics specifications as monads $\mathscr{T}$ on the considered transition category. A specification consists of proof rules for inductively constructing transitions. Intuitively, $\mathscr{T}(X)$ is obtained by saturating its argument X by the considered proof rules, i.e., it augments X with new, formal transitions constructed from the rules. A *$\mathscr{T}$-algebra* is then an object X respecting the rules, formally a map $\mathscr{T}(X) \to X$ satisfying certain conditions.

Now, the crucial result for proving congruence of bisimilarity is that given algebras $a\colon \mathscr{T}(X) \to X$ and $b\colon \mathscr{T}(Y) \to Y$, together with a bisimulation R as above, the saturation $\mathscr{T}(R)$ is again a bisimulation, in the sense that the composite $\mathscr{T}(R) \xrightarrow{\mathscr{T}(f)} \mathscr{T}(X) \xrightarrow{a} X$ satisfies the same lifting property (1) as f above. For this, it suffices to prove that both a and $\mathscr{T}(f)$ do.

First, for a, this means that any commuting square as below should admit a lifting k as shown.



Intuitively, v is a term C with variables $x_1, \ldots, x_n$ in X, which we denote by $v = C[x_1, \ldots, x_n]$, so this says that any α-transition e whose source has the shape $a(C[x_1, \ldots, x_n])$ decomposes as $a(E[e_1, \ldots, e_n])$ for some E and e_i's. Here, E is a transition proof with holes, that are assigned to the e_i's. In other words, transitions of the compound state $a(C[x_1, \ldots, x_n])$ may be described in terms of those of its components $x_1, \ldots, x_n$: this is precisely compositionality!

Let us now explain the standard scheme for proving that $\mathscr{T}(f)$ satisfies (1).

(i) Intuitively, $\mathscr{T}(R)$ consists of terms r with as variables pairs of related elements, i.e.,

$$r = C[(x_1, y_1), \ldots, (x_n, x_n)] \tag{2}$$

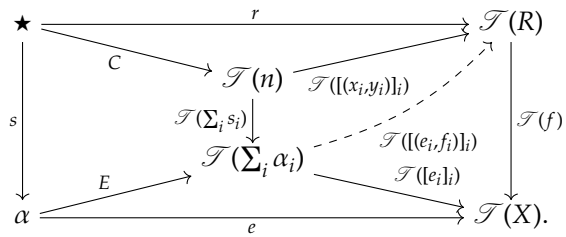
with $(x_i, y_i) \in R$. Or, equivalently, $r = (C[x_1, \ldots, x_n], C[y_1, \ldots, y_n])$.

(ii) The (standard) key insight for proving that this is a bisimulation is that any transition proof $e\colon C[x_1, \ldots, x_n] \xrightarrow{\alpha} x'$ decomposes into a formal α-transition from C, say $E\colon C \xrightarrow{\alpha} D$, and a family of transitions $e_i\colon x_i \xrightarrow{\alpha_i} x_i'$, so that

$$e = E[e_1, \ldots, e_n]. \tag{3}$$

(iii) Indeed, because R is a bisimulation, we then find transitions $f_i\colon y_i \xrightarrow{\alpha_i} y_i'$ with $(x_i', y_i') \in R$, so that $C[y_1, \ldots, y_n] \xrightarrow{\alpha} D[y_1', \ldots, y_n']$ with $(D[x_1', \ldots, x_n'], D[y_1', \ldots, y_n']) \in \mathscr{T}(R)$, as desired.

Familiality provides a high-level language for describing this situation. Indeed, it ensures that any commuting square as the exterior below, decomposes as in the solid part of



(4)

It thus accounts for both decompositions (2) and (3). The lifting property of R as a bisimulation then provides all liftings $[(e_i, f_i)]$, and the composite $\mathscr{T}([(e_i, f_i)]_i) \circ E$ intuitively corresponds to $E[(e_1, f_1), \ldots, (e_n, f_n)]$. This shows that $\mathscr{T}(R)$ is indeed a bisimulation.

Finally, let us mention that our account of bisimulation up to context uses much the same language, except that cannot be directly defined by lifting. This makes it a bit more verbose, so we refrain from exposing it here, and move on to the technical development.

Plan

In §2, relying on the examples of combinatory logic and CCS, we recast structural operational semantics and congruence of bisimilarity in the setting of monads on transition categories. In §3, we prove that compositionality and familiality entail congruence of bisimilarity. In §4, we investigate soundness of bisimulation up to context. In §5, we analyse bisimilarity in the π-calculus using our framework. Finally, we conclude in §6.

Notation and preliminaries

We assume basic familiarity with category theory [13]. For any small category $\mathbb{C}$, we denote by $\widehat{\mathbb{C}}$ the category of *presheaves* on $\mathbb{C}$, i.e., contravariant functors to sets (**Set**) and natural transformations between them. For any $f : c \to c'$ in $\mathbb{C}$ and $X \in \widehat{\mathbb{C}}$ the action $X(f)\colon X(c') \to X(c)$ is denoted by $x \mapsto x \cdot f$. The Yoneda embedding is denoted by $\mathbf{y}\colon \mathbb{C} \to \widehat{\mathbb{C}}$, and often left implicit.

We denote by $el(X)$ the *category of elements* [14] of any presheaf X: it has as objects all pairs (c, x) with $x \in X(c)$, and as morphisms $(c, x) \to (c', x')$ all morphisms $f : c \to c'$ in $\mathbb{C}$ such that $x' \cdot f = x$. We denote the corresponding morphism by $f \upharpoonright x'$. Furthermore, we often abbreviate (c, x) to x.

Finally, we often denote by n the finite set $\{1, \ldots, n\}$.

2 STRUCTURAL OPERATIONAL SEMANTICS SPECIFICATIONS AS MONADS

In this section, we explain how to view labelled transition systems as presheaves over adequate categories, in which bisimulation may be defined by lifting (§2.1), and then show by example how structural operational semantics specifications naturally yield monads (§2.2), an observation originally due to Staton [23], and how to abstractly state congruence of bisimilarity.

2.1 Labelled transition systems as objects in transition categories

Let us first view labelled transition systems as presheaves in concrete examples, and then abstract over what we did and define transition categories and bisimulation therein.

2.1.1 Labelled transition systems. The simplest kind of labelled transition system, the one with just one label, is adequately modelled by **Gph**, the category of (directed, multi) graphs, viewed as presheaves over the category $\star \overset{s}{\underset{t}{\rightrightarrows}} [1]$ (provided one accepts the extra generality of allowing distinct, parallel transitions between nodes). Labels may be accomodated by introduced them as a particular graph, say A, so that labelled transition systems are graphs over A, i.e., morphisms $G \to A$ for some graph G. Furthermore, as is well-known, the slice category $\mathbf{Gph}/A$ is equivalent to $\widehat{el(A)}$, presheaves over the category of elements of A.

E.g., in CCS, labels are elements of $L_{CCS} = \{\tau\} \cup \bigcup_{a \in \mathscr{N}} \{a, \overline{a}\}$, where $\mathscr{N}$ denotes a fixed, infinite set of *names*, for example the natural numbers $\mathbb{N}$. Viewing this as the edge set of a one-vertex graph A_{CCS}, the relevant base category $\mathbb{C}_{CCS} = el(A_{CCS})$ is the category freely generated by the graph with vertices in $\{\star\} \uplus L_{CCS}$, plus, for all $\alpha \in L_{CCS}$, two edges $\star \overset{s}{\longrightarrow} \alpha \overset{t}{\longleftarrow} \star$.

Example 2.1. The simple labelled transition system $x \xleftarrow{\overline{a}} y \underset{b}{\overset{b}{\rightrightarrows}} z \circlearrowright_{a}$ is modelled by the presheaf X with

$$X(\star) = \{x, y, z\} \qquad \begin{array}{c} X(\overline{a}) = \{e\} \\ X(b) = \{f, f'\} \\ X(a) = \{g\} \end{array} \qquad \begin{array}{c} x = e \cdot t \\ y = e \cdot s = f \cdot s = f' \cdot s \\ z = f \cdot t = f' \cdot t = g \cdot s = g \cdot t. \end{array}$$

2.1.2 Bisimulation. Having seen how presheaves on $\mathbb{C}_{CCS}$ model labelled transition systems, let us now explain how to define functional bisimulation by lifting, as sketched in the overview. We will use the following efficient, standard notation [18]. In any category $\mathscr{A}$, a *lifting problem* for morphisms f and g is a commuting square as the solid part below. The lifting problem has a solution when there exists a lifting as shown (dashed) making both triangles commute.

$$\begin{array}{ccc} A & \longrightarrow & C \\ {\scriptstyle f}\downarrow & \nearrow & \downarrow{\scriptstyle g} \\ B & \longrightarrow & D \end{array}$$

Definition 2.2. Let $f \boxslash g$ iff all lifting problems for f and g have at least one solution.

When $f \boxslash g$, we say that g has the *right lifting property* w.r.t. f. We moreover let $f^{\boxslash}$ denote the class of all morphisms that have the right lifting property w.r.t. f. Similarly, for any class $\mathscr{D}$ of morphisms, let $\mathscr{D}^{\boxslash} = \bigcap_{f \in \mathscr{D}} f^{\boxslash}$. We define ${}^{\boxslash}g$ and ${}^{\boxslash}\mathscr{D}$ symmetrically.

Proposition 2.3. *A morphism $f : R \to X$ in $\widehat{\mathbb{C}_{CCS}}$ is a functional bisimulation (according to the standard definition) iff it has the right lifting property w.r.t. all maps of the form $s : \star \to \alpha$ for α a label, or otherwise said, if $f \in \mathbf{T}_s^{\boxslash}$, where $\mathbf{T}_s$ denotes the set of all maps $s : \star \to \alpha$.*

Indeed, a presheaf morphism is automatically a simulation (merely because edges are mapped to edges), and the lifting property says that for any commuting square of the form below left

there is a lifting k as shown that makes both triangles commute. By Yoneda, v denotes a vertex in R, e an α-transition in X, and commutation of the square says that $f(v)$ is the source of e. Existence of k then says that there is an α-transition k in R with source v, mapped by f to e, as above right, which is the standard definition of a functional bisimulation.

General (potentially non-functional) *bisimulations* may be defined as spans $X \xleftarrow{l} R \xrightarrow{r} Y$ where l and r are functional bisimulations. Such bisimulations may have non-monic pairings $R \to X \times Y$, but if we consider their epi-mono factorisation $R \xrightarrow{e} im(R) \xhookrightarrow{m} X \times Y$, we have:

Proposition 2.4. *The monic factor of any bisimulation is again a bisimulation.*

Proof. Consider any factorisation $m \circ e$ as above. Because ★ is representable, its covariant hom-functor preserves epis, i.e., if $f : X \to Y$ is epi, then by Yoneda so is the set map

$$\widehat{\mathbb{C}_{CCS}}(\star, X) \xrightarrow{\widehat{\mathbb{C}_{CCS}}(\star, f)} \widehat{\mathbb{C}_{CCS}}(\star, Y).$$

Thus every lifting problem for any $s\colon \star \to \alpha$ and $im(h) \xhookrightarrow{m} X \times Y \xrightarrow{\pi} X$ induces a lifting problem for s and the composite $\pi \circ m \circ e$. The latter has a lifting by hypothesis, which yields a lifting for $\pi \circ m$. Everything works symmetrically for the projection to Y. □

Definition 2.5. Isomorphism classes of monic bisimulations, i.e., subobjects of $X \times Y$ that are bisimulations, are called *bisimulation relations*.

Finally, for any set I, we may define the union of relations $R_i \hookrightarrow X \times Y$, for $i \in I$, as the image of their (wide) cotupling $\sum_i R_i \to X \times Y$. We then have:

Proposition 2.6. *Bisimulation relations are closed under union, and admit a maximum, called* bisimilarity.

Proof. Because ★ is representable, it is *super tiny*, i.e., its covariant hom-functor preserves epis and colimits. Thus, every lifting problem for any $s\colon \star \to \alpha$ and a given union $\bigcup_i R_i \hookrightarrow X \times Y \xrightarrow{\pi} X$ yields one for some $R_i \hookrightarrow X \times Y \to X$, which has a lifting by hypothesis. Finally, presheaf categories are well-powered, i.e., each object has only a set of subobjects. We thus in particular only have a set of bisimulation relations. This set being closed under unions, it has a maximum. □

2.1.3 Transition categories. Let us now abstract away from the particular example of $\widehat{\mathbb{C}_{CCS}}$. The structure and properties we need for deriving abstract analogues of Propositions 2.4 and 2.6 are:

Definition 2.7. A *transition category* is a cocomplete, finitely complete, well-powered category $\mathscr{A}$ with images, equipped with

- two sets **P** and **L** of objects called *process types* and *label types*, respectively;
- a set **T** of cospans $P \xrightarrow{s} L \xleftarrow{t} Q$ called *transition types*, in which $P, Q \in \mathbf{P}$ and $L \in \mathbf{L}$,

such that process types are *super tiny*, i.e., their covariant hom-functors preserve colimits and epis.

Notation 2.8. We generally denote a transition category by just $\mathscr{A}$, leaving $\mathbf{P}$, $\mathbf{L}$, and $\mathbf{T}$ implicit.

In examples, super tininess follows from:

Remark 2.9. If $\mathscr{A}$ is a presheaf category, then all representable presheaves are super tiny.

Let us now abstractly replay the above development of bisimulation and bisimilarity.

Definition 2.10. For any transition category $\mathscr{A}$, a morphism $f : R \to X$ is a *functional bisimulation* iff it is in $\mathbf{T}_s^{\boxslash}$, where $\mathbf{T}_s$ denotes the class of morphisms appearing as s in some transition type. Given any two objects X and Y, a *bisimulation* is a span $X \xleftarrow{s} R \xrightarrow{t} Y$, or equivalently a map $R \xrightarrow{\langle s,t \rangle} X \times Y$ such that s and t are both functional bisimulations. When the associated pairing $R \to X \times Y$ is monic, we call R a *bisimulation relation.*

Proposition 2.11. *In any transition category $\mathscr{A}$, bisimulations are closed under images, i.e., if $r \colon R \to X \times Y$ is a bisimulation, then so is m in its epi-mono factorisation $R \overset{e}{\twoheadrightarrow} im(R) \overset{m}{\rightarrowtail} X \times Y$.*

Proposition 2.12. *In any transition category, bisimulation relations $R \hookrightarrow X \times Y$ are closed under union, hence admit a maximum, called* bisimilarity *and denoted by $\sim_{X,Y}$, or simply $\sim$ when X and Y are clear from context.*

As expected we have:

Example 2.13 (Graphs). Graphs form a transition category with $\mathbf{P} = \{\mathbf{y}_\star\}$, $\mathbf{L} = \{\mathbf{y}_{[1]}\}$, and the cospan $\star \xrightarrow{s} [1] \xleftarrow{t} \star$ as unique transition type (omitting the Yoneda embedding).

Example 2.14 (CCS labels). The category $\widehat{\mathbb{C}_{CCS}}$ forms a transition category with $\star$ as only process type, all other representables as label types, and all cospans $\star \xrightarrow{s} \alpha \xleftarrow{t} \star$ as transition types.

2.2 Structural operational semantics specifications as monads on transition categories

In the previous section, we have seen by example how labelled transition systems can be viewed as objects in adequate presheaf categories, and how bisimulation can be defined by lifting in this setting. We have then defined transition categories and bisimulation therein, which abstract over this situation. Let us now further explain how structural operational semantics specifications may be viewed as monads on transition categories, again starting with examples and then abstracting away. We then consider congruence of bisimilarity.

2.2.1 First example: combinatory logic. To get a feel for why monads on transition categories are relevant to operational semantics, let us consider the example of combinatory logic, viewed as a labelled transition system on just one label, i.e., a graph.

Let $\mathscr{T}_{CL}$ denote the functor on **Gph** mapping any graph G to the one with

- as vertices all terms generated by the grammar $M, N ::= (\!|v|\!) \mid S \mid K \mid M\ N$, where v ranges over $G(\star)$, S and K are constants, and $M\ N$ stands for the application of a binary symbol (called "application") to M and N;
- edges inductively defined by the following rules, with the given sources and targets.

$$\frac{e \in G(x,y)}{(\!|e|\!) : (\!|x|\!) \to (\!|y|\!)} \qquad \frac{}{s_{M,N,P} : S\ M\ N\ P \to (M\ N)\ (M\ P)} \qquad \frac{}{k_{M,N} : K\ M\ N \to M}$$

$$\frac{L : M \to M'}{L\ N : M\ N \to M'\ N} \qquad \frac{R : N \to N'}{M\ R : M\ N \to M\ N'}$$

The action of $\mathscr{T}_{CL}$ on morphisms is by applying the given graph morphism to vertices $(\!|x|\!)$ and edges $(\!|e|\!)$. This functor is a monad with multiplication (a.k.a. substitution) inductively defined by

$$\begin{array}{rcl} \mu_{G,\star}(\!|m|\!) & = & m \\ \mu_{G,\star}(K) & = & K \\ \mu_{G,\star}(S) & = & S \\ \mu_{G,\star}(M\ N) & = & \mu_{G,\star}(M)\ \mu_{G,\star}(N) \end{array} \qquad \begin{array}{rcl} \mu_{G,[1]}(\!|r|\!) & = & r \\ \mu_{G,[1]}(k_{M,N}) & = & k_{\mu_{G,\star}(M),\mu_{G,\star}(N)} \\ \mu_{G,[1]}(s_{M,N,P}) & = & s_{\mu_{G,\star}(M),\mu_{G,\star}(N),\mu_{G,\star}(P)} \\ \mu_{G,[1]}(L\ N) & = & \mu_{G,[1]}(L)\ \mu_{G,\star}(N) \\ \mu_{G,[1]}(M\ R) & = & \mu_{G,\star}(M)\ \mu_{G,[1]}(R). \end{array}$$

Example 2.15. Let $M = (\!|M_1|\!)\ (\!|M_2|\!)$, with $M_1, M_2 \in \mathscr{T}_{CL}(\varnothing)(\star)$. Then, $\mu_{\varnothing,\star}(M) = M_1\ M_2$. Similarly, for $k_{P,Q} \in \mathscr{T}_{CL}(\varnothing)(K\ P\ Q, P)$, we have $\mu_{\varnothing,[1]}((\!|k_{P,Q}|\!)\ M) = k_{P,Q}\ (M_1\ M_2)$.

Of interest to us is the free algebra $\mathscr{T}_{CL}(\varnothing)$, which is a proof-relevant variant of combinatory logic.

2.2.2 Example with labels: CCS. As a second example useful for illustrating labels, let us deal with CCS. Recalling the base category $\mathbb{C}_{CCS}$ from §2.1.1:

Definition 2.16. Let $\mathscr{T}_{CCS}$ denote the functor on $\widehat{\mathbb{C}_{CCS}}$ such that

- $\mathscr{T}_{CCS}(G)(\star)$ is the set of CCS terms (simplified for expository purposes) with 'process constants' in $G(\star)$, i.e., generated by the grammar $P, Q ::= (\!|x|\!) \mid 0 \mid a.P \mid \overline{a}.P \mid (P|Q) \mid \nu a.P$, with x ranging over $G(\star)$, *not* considered equivalent up to renaming of bound names in $\nu a.P$ (α-equivalence is not necessary for CCS; by contrast, it is necessary for π-calculus, which forces us to move to a more complex base category);
- for all $\alpha \neq \star$, $\mathscr{T}_{CCS}(G)(\alpha)$ is the set of transition proofs $P \xrightarrow{\alpha} Q$, much as in Boudol and Castellani [3], inductively generated by

$$\frac{e \in G(\alpha)}{(\!|e|\!) : (\!|e \cdot s|\!) \xrightarrow{\alpha} (\!|e \cdot t|\!)} \qquad \frac{}{out_P : \overline{a}.P \xrightarrow{\overline{a}} P} \qquad \frac{}{in_P : a.P \xrightarrow{a} P}$$

$$\frac{R : P \xrightarrow{\alpha} Q \qquad \alpha \notin \{a, \overline{a}\}}{\nu a.R : \nu a.P \xrightarrow{\alpha} \nu a.Q} \qquad \frac{L : P \xrightarrow{\alpha} P'}{(L|Q) : (P|Q) \xrightarrow{\alpha} (P'|Q)} (+ \text{ symmetric } Q|L)$$

$$\frac{L : P \xrightarrow{\overline{a}} P' \qquad R : Q \xrightarrow{a} Q'}{L \rhd R : (P|Q) \xrightarrow{\tau} (P'|Q')} (+ \text{ sym. } R \lhd L).$$

Again substitution equips $\mathscr{T}_{CCS}$ with monad structure, and the free algebra $\mathscr{T}_{CCS}(\varnothing)$ is a proof-relevant variant of CCS.

2.2.3 Congruence of bisimilarity, abstractly. In the previous sections, we have seen two example monads on transition categories, which have the labelled transition systems for combinatory logic and CCS as their free algebras. Let us now review what it means for bisimilarity to be a congruence, and then give an abstract definition in the setting of algebras for a monad on a transition category.

Standardly, given a structural operational semantics specification X, we say that bisimilarity is a congruence when for all multi-hole contexts C and pairs $(x_1, y_1), ..., (x_n, y_n)$ of bisimilar processes, we have $C[x_1, ..., x_n] \sim C[y_1, ..., y_n]$.

In the abstract setting, given any monad $\mathscr{T}$ on a transition category $\mathscr{A}$, we may mimick this definition, and even slightly generalise it by considering two different $\mathscr{T}$-algebras:

Definition 2.17. Given a monad $\mathscr{T}$ on a transition category $\mathscr{A}$, and $\mathscr{T}$-algebras $a\colon \mathscr{T}(X) \to X$ and $b\colon \mathscr{T}(Y) \to Y$, we say that bisimilarity (between X and Y) is a *congruence* when the map $\mathscr{T}(\sim_{X,Y}) \to \mathscr{T}(X \times Y) \xrightarrow{\langle \mathscr{T}(\pi), \mathscr{T}(\pi') \rangle} \mathscr{T}(X) \times \mathscr{T}(Y) \xrightarrow{a \times b} X \times Y$ factors through $\sim_{X,Y} \hookrightarrow X \times Y$.

3 CONGRUENCE OF BISIMILARITY

In the previous sections, we have explained in which sense monads on transition categories model structural operational semantics specifications, and defined what it means for bisimilarity to be a congruence in this setting. We now turn to proving it, by abstracting away each step of the standard proof method sketched in the overview. We first consider compositionality in §3.1. Steps *(i)* and *(ii)* are then dealt with by familiality in §3.2. Finally, in §3.3, Step *(iii)* requires us to refine standard familiality into $\mathbf{T}_s$-familiality. Relying on the theory of weak factorisation systems, we are then able to prove congruence of bisimilarity.

3.1 Compositionality

In the overview, our proof sketch for congruence of bisimilarity started by using compositionality of the considered algebra. Compositionality is easy to express in the abstract framework:

Definition 3.1. A $\mathscr{T}$-algebra $a\colon \mathscr{T}(X) \to X$ is *compositional* iff $a \in \mathbf{T}_s^{\boxslash}$.

Concretely, this says that given any square

$$\begin{array}{ccc} P & \xrightarrow{p} & \mathscr{T}(X) \\ {\scriptstyle \mathbf{T}_s \ni s}\downarrow & \overset{k}{\nearrow} & \downarrow{\scriptstyle a} \\ L & \xrightarrow[r]{} & X \end{array}$$

there exists a lifting k making both triangles commute. Thinking as above of p as a process with variables in X, i.e., a context applied to some processes in X, of $a \circ p$ as its evaluation in X, and of r as a transition from $a \circ p$, this says that r may be decomposed as the evaluation $a \circ k$ of some transition proof k with variables in X and domain $k \circ s = p$.

Proposition 3.2. *The free $\mathscr{T}_{CL}$-algebra $\mathscr{T}_{CL}(\varnothing)$ is compositional, i.e., $\mu_\varnothing \in \mathbf{T}_s^{\boxslash}$.*

Proof sketch (more detail in §A). Any $M \in \mathscr{T}_{CL}^2(\varnothing)$ is a process with variables in $\mathscr{T}_{CL}(\varnothing)$, i.e., it has the shape $C[M_1, ..., M_n]$. We need to be able to decompose any transition $R\colon \mu_{\varnothing,\star}(M) \to N$ into $E[e_1, ..., e_n]$ such that $\mu_{\varnothing,[1]}(E[e_1, ..., e_n]) = R$. We proceed by induction on C. E.g., if $C = C_1\ C_2$ and $R = LP$, then there exists $i \in n$ such that $L \cdot s = \mu_{\varnothing,\star}(C_1[M_1, ..., M_{i-1}])$ and $P = \mu_{\varnothing,\star}(C_2[M_i, ..., M_n])$. Then, by induction hypothesis, we find $E'[e'_1, ..., e'_{i-1}]$ such that $\mu_{\varnothing,[1]}(E'[e'_1, ..., e'_{i-1}]) = L$, so we can pick $E = E'\ C_2$. □

Proposition 3.3. *The $\mathscr{T}_{CCS}$-algebra $\mathscr{T}_{CCS}(\varnothing)$ is compositional.*

Proof. Similar to $\mathscr{T}_{CL}$, using the fact that, because we do not mod out by α-equivalence, the *va* operator may be considered as a unary operator indexed by names. □

3.2 Familiality

Let us now turn to proving that $\mathscr{T}(f) \in \mathbf{T}_s^{\boxslash}$. In the overview, we announced that our treatment of Steps *(i)* and *(ii)* would rely on familial functors. Indeed, what we need is a canonical decomposition of terms $p\colon P \to \mathscr{T}(X)$ and transitions $e\colon L \to \mathscr{T}(X)$ into a context followed by an assignment of its holes. The correct requirement is that p should decompose as

$$P \xrightarrow{C} \mathscr{T}(A) \xrightarrow{\mathscr{T}(h)} \mathscr{T}(X), \tag{5}$$

where C is *generic*, in the sense of Weber [25]. Existence of decompositions (5) with generic C is precisely the familiality of the title, which originates in *familially representable* functors [4], as developed by Weber [25] under the name of *parametric right adjoints*, here called *familial* functors following Garner and Hirschowitz [8].

Definition 3.4. Given any functor $\mathscr{F} : \mathscr{A} \to \mathscr{X}$, a morphism $\xi \colon X \to \mathscr{F}(A)$ is *$\mathscr{F}$-generic*, or *generic* for short, when for all commuting squares of the form below

$$\begin{array}{ccc} X & \xrightarrow{\chi} & \mathscr{F}(B) \\ {\scriptstyle\xi}\downarrow & \overset{\mathscr{F}(h)}{\nearrow} & \downarrow{\scriptstyle\mathscr{F}(g)} \\ \mathscr{F}(A) & \xrightarrow[\mathscr{F}(f)]{} & \mathscr{F}(C) \end{array} \tag{6}$$

there exists a unique h such that $\mathscr{F}(h) \circ \xi = \chi$ and $g \circ h = f$.

Definition 3.5. A functor $\mathscr{F}$ is *familial* when any morphism $X \to \mathscr{F}(A)$ factors as the composite of some generic morphism followed by a *free* one, i.e., one of the form $\mathscr{F}(f)$. A monad $(\mathscr{T}, \eta, \mu)$ is *familial* when the underlying endofunctor is, and furthermore η and μ are *cartesian* natural transformations, i.e., all their naturality squares are pullbacks.

Remark 3.6. By the pullback lemma, when the domain category has a terminal object, a natural transformation $\alpha \colon F \to G$ is cartesian iff its naturality squares of the form below are pullbacks.

$$\begin{array}{ccc} FA & \xrightarrow{F(!)} & F1 \\ {\scriptstyle\alpha_A}\downarrow & & \downarrow{\scriptstyle\alpha_1} \\ GA & \xrightarrow[G(!)]{} & G1 \end{array}$$

In such a decomposition (5), A should be thought of as representing the holes of C, and h as assigning to each hole the corresponding process. Similarly, of course, each transition $t \colon L \to \mathscr{T}(X)$ should factor into $L \xrightarrow{D} \mathscr{T}(R) \xrightarrow{\mathscr{T}(k)} \mathscr{T}(X)$ with generic D, and the mediating arrow $\sum_i s_i$ in (4) follows from genericity of C.

Example 3.7. On **Set**, if the considered functor $\mathscr{T}$ is finitary, then A is finite, so we may simply choose it to be the ordinal corresponding to the number n_C of holes in C. Familial functors thus coincide with standard polynomial functors [12], as we have $\mathscr{T}(X) \cong \sum_C X^{n_C}$.

There is a slight generalisation of the formula of Example 3.7 to presheaf categories, which will be useful for showing that the monads $\mathscr{T}_{CL}$ and $\mathscr{T}_{CCS}$ of §2.2.1 and §2.2.2 are familial:

Lemma 3.8 ([25, Remark 2.12]). *An endofunctor $\mathscr{T}$ on any presheaf category $\widehat{\mathbb{C}}$ is familial iff there is a functor $E \colon el(\mathscr{T}(1)) \to \widehat{\mathbb{C}}$ and a natural isomorphism (in X and c):*

$$\mathscr{T}(X)(c) \cong \sum_{x \in \mathscr{T}(1)(c)} \widehat{\mathbb{C}}(E(c,x), X).$$

Proof sketch. In presheaf categories, familiality is equivalent to *pointwise familiality*, i.e., existence of a generic-free factorisation for all morphisms of the form $\mathbf{y}_c \to \mathscr{T}(X)$. Any functor of the given form is clearly pointwise familial, for any map $(x, \varphi) \colon \mathbf{y}_c \to \mathscr{T}(X)$, with $\varphi \colon E(c,x) \to X$, factors as $\mathbf{y}_c \xrightarrow{(x,id)} \mathscr{T}(E(c,x)) \xrightarrow{\mathscr{T}(\varphi)} \mathscr{T}(X)$. Conversely, if $\mathscr{T}$ is pointwise familial, define $E(c,x)$ for any $x \colon \mathbf{y}_c \to \mathscr{T}(1)$ to be given by (any global choice of) generic-free factorisation of $x \colon \mathbf{y}_c \to \mathscr{T}(E(c,x)) \to \mathscr{T}(1)$. □

Proposition 3.9. *The monad $\mathscr{T}_{CL}$ is familial.*

Proof sketch (more detail in §A). By Lemma 3.8, it suffices to exhibit a functor $E\colon el(\mathscr{T}_{CL}(1)) \to \mathbf{Gph}$, such that $\mathscr{T}_{CL}(Z)(c) \cong \sum_{x\in\mathscr{T}_{CL}(1)(c)}[E(c,x), Z]$, naturally in c and Z. Now, $\mathscr{T}_{CL}(1)(\star)$ consists of terms on a unique free variable, say $\top$, and we define E to map any such term C to the discrete graph with vertices in the ordinal n_C, where n_C is the number of occurrences of $\top$ in C. Similarly, $\mathscr{T}_{CL}[1]$ consists of transition derivations on just one transition axiom, say $\Tau\colon \top \to \top$. On such derivations, we define E by induction:

$$E(\Tau) = \mathbf{y}_{[1]}, \qquad E(k_{M,N}) = E(M) + E(N), \qquad E(L\,N) = E(L) + E(N),\ldots$$

E is then defined on $s \upharpoonright R$ and $t \upharpoonright R$, by straightforward induction. □

Proposition 3.10. *$\mathscr{T}_{CCS}$ is familial.*

Proof. Similar, using the fact that we do not mod out by α-equivalence. □

3.3 Congruence of bisimilarity and $\mathbf{T}_s$-familiality

Let us now get to the final Step *(iii)* of our proof of congruence of bisimilarity. In the overview, we obtained by familiality of $\mathscr{T}$ the mediating morphism $\sum_i s_i$, whose very particular form allowed us to find a lifting: we found one for each s_i individually, and then took the cotupling. In the general case, however, there is *a priori* no guarantee that the mediating morphism will have such a nice form. Now, the only thing we need in order to conclude is that the mediating morphism be in ${}^{\boxtimes}(\mathbf{T}_s^{\boxtimes})$, so it is tempting to take this as an additional hypothesis. But the question is then whether this will be expressive enough to cover our examples. E.g., will ${}^{\boxtimes}(\mathbf{T}_s^{\boxtimes})$ always contain coproduct of maps in $\mathbf{T}_s$ and isomorphisms? This is where basic results from weak factorisation systems [9, 18] come to the rescue.

3.3.1 Weak factorisation systems and ${}^{\boxtimes}(\mathbf{T}_s^{\boxtimes})$. Indeed, the point is that the classes of maps ${}^{\boxtimes}(\mathbf{T}_s^{\boxtimes})$ and $\mathbf{T}_s^{\boxtimes}$ will form a *cofibrantly generated weak factorisation system*. Let us start with the most general notion:

Definition 3.11. A *weak factorisation system* on a category $\mathscr{A}$ consists of two classes of maps $\mathscr{L}$ and $\mathscr{R}$ such that $\mathscr{L}^{\boxtimes} = \mathscr{R}$, $\mathscr{L} = {}^{\boxtimes}\mathscr{R}$, and every map $f\colon A \to B$ factors as $A \xrightarrow{l} C \xrightarrow{r} B$ with $l \in \mathscr{L}$ and $r \in \mathscr{R}$.

Cofibrantly generated weak factorisation systems are those generated from a set of maps by lifting (this is the so-called *small object* argument). We introduce them in Proposition 3.15 below, which requires us to first define *transfinite* composition, *small* objects, and *relative cell complexes*:

Definition 3.12. For any ordinal λ, a *λ-sequence* is a cocontinuous functor from λ viewed as a category, to $\mathscr{A}$. A *transfinite composite* of any λ-sequence $X\colon \lambda \to \mathscr{A}$ is the component $\rho_0\colon \lambda(0) \to colim_{\beta<\lambda}X(\beta)$ of any colimiting cocone ρ.

Definition 3.13. Let $\mathscr{J}$ denote any class of morphisms in a cocomplete category $\mathscr{A}$, and let κ denote any cardinal. An object $A \in \mathscr{A}$ is *κ-small* relative to $\mathscr{J}$ iff, for all κ-filtered [9, Definition 2.1.12] ordinals λ and λ-sequences $X\colon \lambda \to \mathscr{A}$ such that $X(\beta) \to X(\beta+1)$ is in $\mathscr{J}$ for all $\beta+1 < \lambda$, the canonical map $colim_{\beta<\lambda}\mathscr{A}(A, X(\beta)) \to \mathscr{A}(A, colim_{\beta<\lambda}X(\beta))$ is bijective. We say that A is *small relative to* $\mathscr{J}$ iff it is κ-small relative to $\mathscr{J}$ for some κ.

Definition 3.14. For any class $\mathscr{J}$ of maps in a cocomplete category $\mathscr{A}$, let *$\mathscr{J}$-cell* denote the class of transfinite composites of pushouts of maps in $\mathscr{J}$, which we call *relative $\mathscr{J}$-cell complexes*.

Proposition 3.15 ([9, Theorem 2.1.14]). *For any set $\mathscr{J}$ of maps in a cocomplete category, if the domains of maps in $\mathscr{J}$ are small relative to $\mathscr{J}$-cell, then $({}^{\boxslash}\mathscr{J})^{\boxslash}$ and $\mathscr{J}^{\boxslash}$ form a weak factorisation system. Any so obtained weak factorisation system is called* cofibrantly generated.

This applies to our abstract setting:

Proposition 3.16. *In any transition category, process types are small relative to $\mathbf{T}_s$-cell.*

Proof. Smallness of an object A means precisely that its covariant hom-functor preserves certain transfinite compositions, which holds for process types by super tininess. □

We thus have by Proposition 3.15:

Corollary 3.17. *In any transition category, ${}^{\boxslash}(\mathbf{T}_s^{\boxslash})$ and $\mathbf{T}_s^{\boxslash}$) form a weak factorisation system.*

The good thing with cofibrantly generated weak factorisation systems is that they enjoy an explicit characterisation of $\mathscr{L}$, which we now recall.

Definition 3.18. A *retract* of $f : X \to Y$ is any map $g\colon A \to B$ for which there exists a retraction $f \to g$ in the arrow category $\mathscr{A}^{\to}$, i.e., morphisms $g \xrightarrow{s} f \xrightarrow{r} g$ such that $r \circ s = id_g$.

Proposition 3.19 ([9, Corollary 2.1.15]). *In the setting of Proposition 3.15, the left class $({}^{\boxslash}\mathscr{J})^{\boxslash}$ consists precisely of retracts of relative $\mathscr{J}$-cell complexes.*

We thus have:

Corollary 3.20. *In any transition category, the classes of maps $({}^{\boxslash}(\mathbf{T}_s^{\boxslash}), \mathbf{T}_s^{\boxslash})$ form a factorisation system whose left class consists precisely of retracts of relative $\mathbf{T}_s$-cell complexes. In particular, ${}^{\boxslash}(\mathbf{T}_s^{\boxslash})$ contains coproducts of maps in $\mathbf{T}_s$ and of isomorphisms.*

3.3.2 Congruence of bisimilarity. Being assured that ${}^{\boxslash}(\mathbf{T}_s^{\boxslash})$ is large enough for our purposes, let us now return to congruence of bisimilarity.

Definition 3.21. A monad $(\mathscr{T}, \eta, \mu)$ on a transition category $\mathscr{A}$ is *$\mathbf{T}_s$-familial* when it is familial and furthermore for any commuting diagram

$$\begin{array}{ccc} P & \xrightarrow{\quad s \quad} & L \\ {\scriptstyle C}\downarrow & & \downarrow{\scriptstyle D} \\ \mathscr{T}(A) & \xrightarrow[\mathscr{T}(s')]{} & \mathscr{T}(R) \end{array}$$

where $s \in \mathbf{T}_s$ and C and D are generic, then $s' \in {}^{\boxslash}(\mathbf{T}_s^{\boxslash})$.

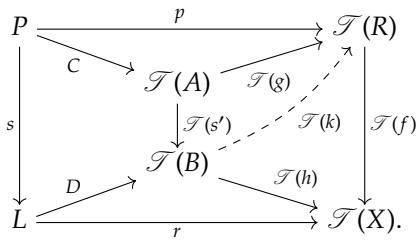
Remark 3.22. By Lemma 3.8, when $\mathscr{A}$ is a presheaf category and P and L are representable, this is equivalent to requiring that all maps of the form $s' \cong E(s \upharpoonright x)$ (for some $x \in \mathscr{T}(1)(L)$) are in ${}^{\boxslash}(\mathbf{T}_s^{\boxslash})$. Indeed, above, take for x the composite $L \xrightarrow{D} \mathscr{T}(R) \xrightarrow{\mathscr{T}(!)} \mathscr{T}(1)$: we then have $R \cong E(L, x)$, $A \cong E(P, x \cdot s)$, and $s' \cong E(s \upharpoonright x)$.

Theorem 3.23. *For any compositional algebra $a\colon \mathscr{T}(X) \to X$ for a $\mathbf{T}_s$-familial monad $\mathscr{T}$ on a transition category, if $f : R \to X$ is a functional bisimulation, so is $\mathscr{T}(R) \xrightarrow{\mathscr{T}(f)} \mathscr{T}(X) \xrightarrow{a} X$.*

For proving this, let us appeal to standard closure properties of weak factorisation systems:

Proposition 3.24 ([18, Lemma 11.1.4]). *For any factorisation system $(\mathscr{L}, \mathscr{R})$, $\mathscr{L}$ (resp. $\mathscr{R}$) contains all isomorphisms and is closed under composition, retracts, coproducts (resp. products) of arrows, and pushouts (resp. pullbacks). $\mathscr{L}$ is furthermore closed under transfinite composition.*

Proof of Theorem 3.23. By Proposition 3.24, functional bisimulations are closed under composition, so it suffices to show that $\mathscr{T}(f)$ is one. We thus need to construct a lifting for any commuting square as the exterior of



We use familiality of $\mathscr{T}$ to factor p as $\mathscr{T}(g)\circ C$ and r as $\mathscr{T}(h)\circ D$ with C and D generic. Genericity of C then yields $s'\colon A \to B$ as shown, which is in ${}^{\boxslash}(\mathbf{T}_s^{\boxslash})$ by $\mathbf{T}_s$-familiality, so we obtain a lifting k as shown, and $\mathscr{T}(k)\circ D$ is a lifting for the whole diagram. □

Corollary 3.25. *For any bisimulation $R \hookrightarrow X \times Y$ between compositional algebras for a $\mathbf{T}_s$-familial monad $\mathscr{T}$, the induced map $\mathscr{T}(R) \to X \times Y$ is a bisimulation.*

Corollary 3.26. *Between any two compositional algebras for a $\mathbf{T}_s$-familial monad $\mathscr{T}$, bisimilarity is a congruence.*

In most examples, the considered algebra is the free one $\mathscr{T}(\varnothing)$. This is thus covered by:

Corollary 3.27. *Consider any $\mathbf{T}_s$-familial monad $\mathscr{T}$ on some transition category $\mathscr{A}$ such that $\mu_1\colon \mathscr{T}^2(1) \to \mathscr{T}(1)$ is a functional bisimulation. Then for all X, if $f : R \to \mathscr{T}(X)$ is a functional bisimulation, so is $\mathscr{T}(R) \xrightarrow{\mathscr{T}(f)} \mathscr{T}^2(X) \xrightarrow{\mu_X} \mathscr{T}(X)$, and hence bisimilarity in $\mathscr{T}(X)$ is a congruence.*

Proof. Because $\mathscr{T}$ is familial, all naturality squares for μ are pullbacks, so μ_X is a functional bisimulation by Proposition 3.24. We conclude by the theorem. □

The case of §5.2 is an exception: the π-calculus *is* a free algebra for a certain monad $\mathscr{T}_\pi$, but considered as an algebra for a certain submonad of $\mathscr{T}_\pi$, and as such not free.

4 BISIMULATION UP TO CONTEXT

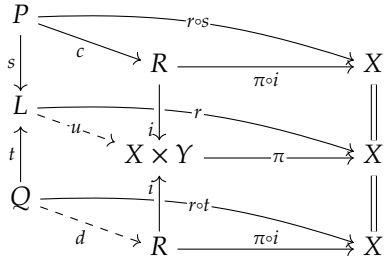
In this section, we consider an alternative notion of bisimulation in transition categories, *pre-bisimulations*, which we relate to bisimulations by showing that under a mild additional hypothesis any pre-bisimulation embeds into some bisimulation. We then define a notion of *pre-bisimulation up to context*, which in examples corresponds to bisimulation up to context. Finally, we prove that pre-bisimulation up to context is sound, in the sense that any pre-bisimulation up to context embeds into some pre-bisimulation (and hence into some bisimulation).

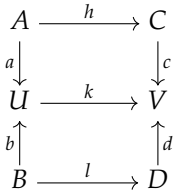
4.1 Pre-bisimulations

In Definition 2.10, we defined functional bisimulations through a lifting property, which is flexible enough to, e.g., exclude some transitions from the considered labelled transition systems X and Y. The following is in fact closer to the ordinary definition of a bisimulation.

Definition 4.1. Consider any transition category. A *pre-bisimulation* is a map $i\colon R \to X \times Y$ such that for all transition types $P \xrightarrow{s} L \xleftarrow{t} Q$, processes $c\colon P \to R$, and transitions $r\colon L \to X$ making

the solid part below left commute, there exist u and d as shown making the whole commute, and symmetrically for Y.

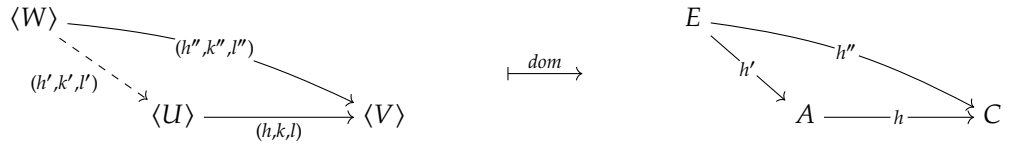




(7)

Equivalently, the pre-bisimulation condition may be formulated as a *weak cartesianness* property of the cospan map formed by the front face above left, relative to transition types. Indeed, consider the category $\mathscr{A}^\vee$ with cospans in $\mathscr{A}$ as objects, and as morphisms all commuting diagrams of the form above right. Taking the top component (i.e., sending (h,k,l) to h) defines a functor $dom\colon \mathscr{A}^\vee \to \mathscr{A}$.

Definition 4.2. A morphism as to the right of (7) is *weakly cartesian* relative to some cospan $E \to W \leftarrow F$ iff, denoting by $\langle U\rangle$ the cospan with U as middle component (and inferring the rest from context), for all morphisms $(h'',k'',l'')\colon \langle W\rangle \to \langle V\rangle$ in $\mathscr{A}^\vee$, and morphisms $h'\colon E \to A$ such that $h \circ h' = h''$, there exists k' and l' making the diagram below commute.



Remark 4.3. This is cartesianness is the sense of Grothendieck fibrations, except with weakened universal property: it is restricted to certain cospans, and the mediating arrow need not be unique.

4.2 From pre-bisimulation to bisimulation

Let us now relate pre-bisimulation to bisimulation. Intuitively, the value of R over transitions is irrelevant in the definition of pre-bisimulation. But, thinking of R as a relation, one would expect that by completing it with all transitions that exist in $X \times Y$ between pairs of related processes we would get a bisimulation. This completion operation may be performed generically by a factorisation system, up to an additional stratification hypothesis. Let us first define the factorisation system in question, then introduce the needed hypothesis, and finally relate pre-bisimulation to bisimulation (Proposition 4.8).

Proposition 4.4. *In any transition category $\mathscr{A}$, the set $\mathscr{J} = \{[s,t]\colon P + Q \to L \mid (s,t) \in \mathbf{T}\}$ generates a weak factorisation system, say $(\mathscr{L},\mathscr{R})$.*

Proof. By Proposition 3.15 and tininess of process types. □

Definition 4.5. A transition category $\mathscr{A}$ is *two-level* iff $\mathscr{J} \subseteq \mathbf{P}^{\boxslash}$, viewing each $P' \in \mathbf{P}$ as the unique map $0 \to P'$.

Explicitly, $\mathscr{A}$ is two-level when any map $f\colon P' \to L$ with $P' \in \mathbf{P}$ lifts through any $[s,t]$ with codomain L, i.e., there exists k making the following triangle commute.

$$\begin{array}{ccc} & & P+Q \\ & \overset{k}{\nearrow} & \downarrow{\scriptstyle [s,t]} \\ P' & \xrightarrow[f]{} & L \end{array} \tag{8}$$

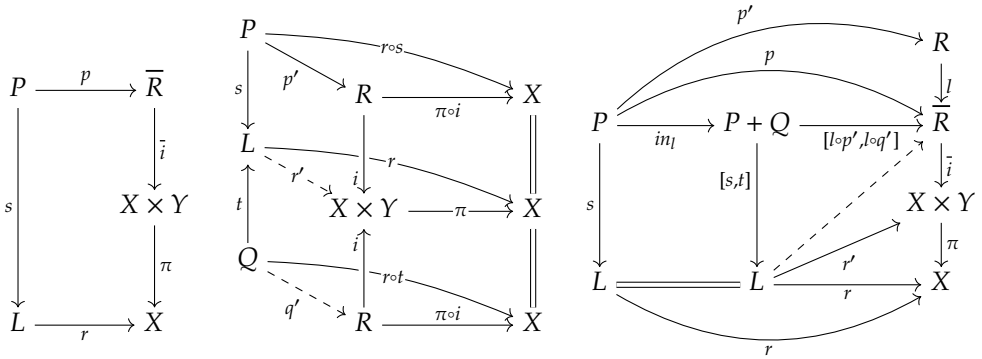
Remark 4.6. All our example transition categories are two-level.

Lemma 4.7. *In any two-level transition category, we have* $\mathscr{L} \subseteq \mathbf{P}^{\boxslash}$.

Proof. By Proposition 3.19, it suffices to show that $\mathbf{P}^{\boxslash}$ contains $\mathscr{J}$ and is stable under pushout, transfinite composition, and retracts. First of all, $\mathscr{J} \subseteq \mathbf{P}^{\boxslash}$ holds by hypothesis. Stability under retracts is direct, and stability under pushout and transfinite composition follows from tininess of process types (by transfinite induction in the latter case). □

Proposition 4.8. *For any pre-bisimulation* $i\colon R \to X \times Y$ *in a two-level transition category, the factor* $\bar{i}$ *of the* $(\mathscr{L}, \mathscr{R})$*-factorisation* $R \overset{l}{\longrightarrow} \overline{R} \overset{\bar{i}}{\longrightarrow} X \times Y$ *of* i *is a bisimulation.*

Proof. Consider any commuting square as below left. By Lemma 4.7, we find a lifting, say $p'\colon P \to R$ of p through $l\colon R \to \overline{R}$. Because R is a pre-bisimulation, we get r' and q' making the diagram below center commute. We may thus factor the left-hand square as below right, and hence get the desired (dashed) lifting by $\bar{i} \in \mathscr{R}$.



□

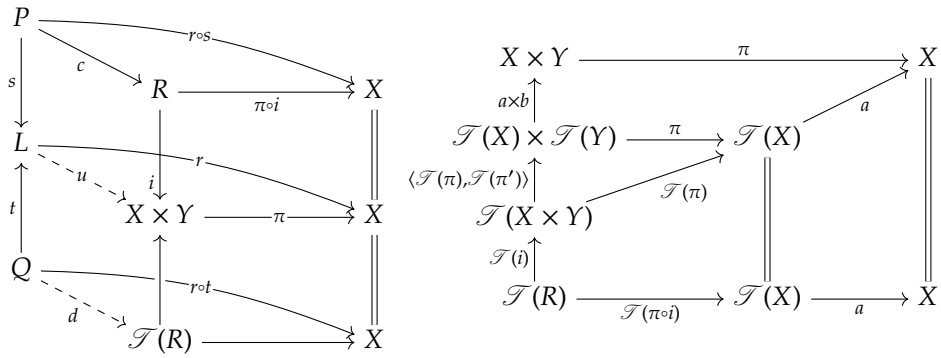
4.3 Pre-bisimulation up to context

After defining pre-bisimulations in the previous section, and showing that they embed into bisimulations by factorisation, we now proceed in this section to defining pre-bisimulations up to context, and showing that they embed into pre-bisimulations, hence into bisimulations, provided an additional saturation hypothesis holds.

Let $\mathscr{T}$ denote a familial monad on a transition category $\mathscr{A}$, and let $a\colon \mathscr{T}(X) \to X$ and $b\colon \mathscr{T}(Y) \to Y$ be $\mathscr{T}$-algebras.

Definition 4.9. A map $i\colon R \to X \times Y$ is a *pre-bisimulation up to* $\mathscr{T}$ iff for all transition types $P \overset{s}{\longrightarrow} L \overset{t}{\longleftarrow} Q$, processes $c\colon P \to R$, and transitions $r\colon L \to X$ making the solid part below left commute, there exist u and d as shown making the whole commute, and symmetrically for Y,

where the bottom square is defined on the right.

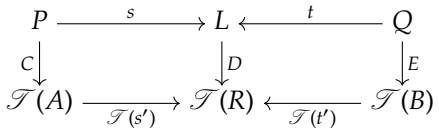


(9)

Following Definition 4.2, this may be expressed as weak cartesianness of the front face relative to transition types.

Before proving soundness of pre-bisimulation up to $\mathscr{T}$, we need to refine $\mathbf{T}_s$-familiality. Indeed, in order to find the desired lifting in the proof of Theorem 3.23, we needed to assume that the image by E of any $s \in \mathbf{T}_s$ was in ${}^{\boxtimes}(\mathbf{T}_s^{\boxtimes})$. With pre-bisimulations, things are different, because they are not defined by lifting. Instead, we thus use the following notion of saturated pre-bisimulation up to $\mathscr{T}$.

Definition 4.10. Let $\overline{\mathbf{T}}$ denote the class of cospans (s', t') occurring in any diagram of the form below, where $(s, t) \in \mathbf{T}$ and C, D, and E are generic.

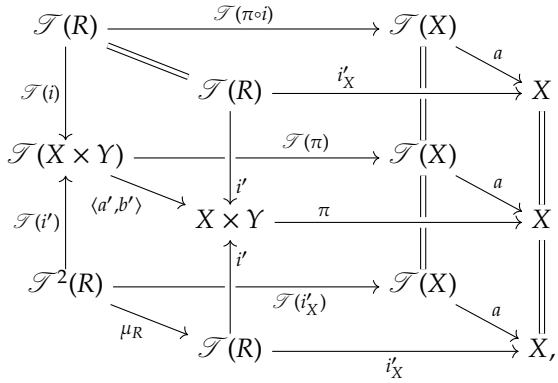


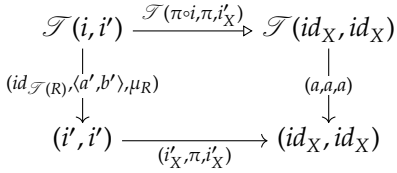
A pre-bisimulation up to $\mathscr{T}$ is *saturated* iff the front face to the left of (9) is weakly cartesian relative to all cospans in $\overline{\mathbf{T}}$.

Theorem 4.11. *For any familial monad $\mathscr{T}$, compositional $\mathscr{T}$-algebras $a\colon \mathscr{T}(X) \to X$ and $b\colon \mathscr{T}(Y) \to Y$, and saturated pre-bisimulation $i\colon R \to X \times Y$ up to $\mathscr{T}$, the following map is a pre-bisimulation:*

$$\mathscr{T}(R) \xrightarrow{\mathscr{T}(i)} \mathscr{T}(X \times Y) \xrightarrow{\langle \mathscr{T}(\pi), \mathscr{T}(\pi') \rangle} \mathscr{T}(X) \times \mathscr{T}(Y) \xrightarrow{a \times b} X \times Y. \tag{10}$$

Proof. Let a' denote the map $\mathscr{T}(X \times Y) \xrightarrow{\mathscr{T}(\pi)} \mathscr{T}(X) \xrightarrow{a} X$ and define b' symmetrically. Then, let $i'_X\colon \mathscr{T}(R) \to X$ denote the map $\mathscr{T}(R) \xrightarrow{\mathscr{T}(i)} \mathscr{T}(X \times Y) \xrightarrow{a'} X$, and define $i'_Y\colon \mathscr{T}(R) \to Y$ symmetrically. Then, letting i' denote the map (10), we have $i' = \langle i'_X, i'_Y \rangle$, $i'_X = \pi \circ i'$, and $i'_Y = \pi' \circ i'$,... Now, the cube below left commutes (notably using the monad algebra laws), which means that it forms a square as below right in $\mathscr{A}^{\vee}$,

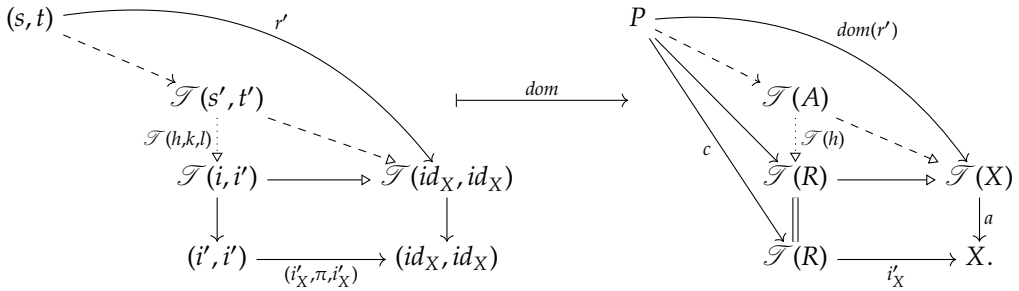




whose top arrow is precisely the componentwise image by $\mathscr{T}$ of the front face to the left of (9).

Notation 4.12. Open arrow heads denote arrows of the form $\mathscr{T}(-)$, and commutation of a diagram of such arrows means that the underlying arrows, without $\mathscr{T}$, agree.

By symmetry, it suffices to prove that the bottom map (i'_X, π, i'_X) is weakly cartesian relative to transition types. . Let us thus consider a map $r\colon (s,t) \to (id_X, id_X)$, where $(s,t) \in \mathbf{T}$ denotes any transition type, whose top component $dom(r)$ factors through $dom(i'_X, \pi, i'_X) = i'_X$, say as $P \xrightarrow{c} \mathscr{T}(R) \xrightarrow{i'_X} X$. By compositionality of X, r factors as $(s,t) \xrightarrow{r'} \mathscr{T}(id_X, id_X) \xrightarrow{a} (id_X, id_X)$, so we get a diagram like the solid part of



By familiality of $\mathscr{T}$, we get a componentwise generic-free factorisation of r', shown dashed below. By genericity of its top component, we get a morphism h as shown. Finally, by weak cartesianness of $(i,i') \to (id_X, id_X)$ relative to $\overline{\mathbf{T}}$, we obtain a morphism (h,k,l) as shown, which yields the desired lifting, namely the composite $(s,t) \to \mathscr{T}(s',t') \xrightarrow{\mathscr{T}(h,k,l)} \mathscr{T}(i,i') \to (i',i')$. □

Corollary 4.13. *For any familial monad $\mathscr{T}$ and compositional $\mathscr{T}$-algebras $a\colon \mathscr{T}(X) \to X$ and $b\colon \mathscr{T}(Y) \to Y$, any saturated pre-bisimulation up to $\mathscr{T}$ embeds into some bisimulation.*

Proof. By Theorem 4.11 and Proposition 4.8. □

In order to easily apply the theorem, although we do not have any precise characterisation like that of ${}^{\boxtimes}(\mathbf{T}^{\boxtimes}_s)$, we do have:

Proposition 4.14. *If $\overline{\mathbf{T}}$ consists (up to isomorphism) of coproducts of isomorphisms and cospans in $\mathbf{T}$, then all pre-bisimulations up to $\mathscr{T}$ are saturated.*

Example 4.15. As an example application, bisimulation up to context is sound for combinatory logic and CCS.

5 THREE SHADES OF π-CALCULUS

Let us now consider a more significant example than combinatory logic and CCS: the π-calculus. Unlike in CCS, we have to mod out by α-equivalence, because channel names may be input, hence substituted deep in process terms, which may force renaming. We essentially follow the presentation of Sangiorgi and Walker [21, §1.3], using a simplified variant for expository purposes.

In §5.1, we make a first attempt at covering π-calculus using our approach. We manage to design a familial monad, $\mathscr{T}_\pi$, over a certain presheaf category $\widehat{\mathbb{B}}$, which faithfully encodes the desired labelled transition system. However, as bisimilarity is known not to be a congruence in π, something is bound to fail. And indeed, we show that the initial $\mathscr{T}_\pi$-algebra $\mathscr{T}_\pi(\varnothing)$ of processes is not compositional. We rectify this in a standard way in §5.2, by defining a familial submonad, $\mathscr{T}_\pi^-$, such that the $\mathscr{T}_\pi^-$-algebra $\mathscr{T}_\pi(\varnothing)$ is compositional, thus recovering the known facts that (1) standard bisimilarity is a congruence for all operators but input, and (2) bisimulation up to non-input context is sound. We finally consider in §5.3 a different, though still standard, way of remedying the non-congruence problem. This consists in restricting attention to relations, called *wide open* [7, 19, 23] that are stable under channel renaming. We do this by working over a different base category $\mathbb{F}$, and adapting the definition of $\mathscr{T}_\pi$, yielding a new familial monad $\mathscr{T}_\pi^+$. We recover the known results that wide-open bisimilarity is a congruence and that wide-open bisimulation up to context is sound.

5.1 Basic approach

In this section, we illustrate our approach by examining the failure of congruence for standard bisimilarity in π. We analyse the problem, which allows us to consider two different solutions in the next two sections.

Naively adaptating what we did with CCS to the π-calculus, we could try considering a similar base category with CCS labels replaced with π-calculus labels, $\tau, o_{a,b}, \iota_{a,b}$, etc. However, this setting cannot accomodate the input axiom: $a(c).P \xrightarrow{\iota_{a,b}} P[c \mapsto b]$. Indeed, this requires a definition of renaming, for which a standard, inductive definition stumbles upon the base case: it does not make any sense to replace c with b in $(\!|x|\!)$. We thus consider a different base category.

Definition 5.1. Let $\mathbb{B}$ denote the subcategory of $\mathbf{Set}^{op}$ with finite subsets of $\mathscr{N}$ as objects and bijections as morphisms, augmented with

- objects $\tau_\gamma, o_{\gamma,a,b}, o^\nu_{\gamma,a,c}, \iota_{\gamma,a,b}$, and $\iota^\nu_{\gamma,a,c}$ for all $\gamma \in \mathscr{P}_f(\mathscr{N})$, $a, b \in \gamma$, and $c \notin \gamma$,
- morphisms, arranged by transition types (denoting by γ, c the (disjoint) union $\gamma \uplus \{c\}$ – here but not, e.g., in $o_{\gamma,a,b}$: input and output labels have three arguments, a γ and two names; we rely on context to disambiguate):

$$\gamma \xrightarrow{s} \tau_\gamma \xleftarrow{t} \gamma \quad \gamma \xrightarrow{s} o_{\gamma,a,b} \xleftarrow{t} \gamma \quad \gamma \xrightarrow{s} \iota_{\gamma,a,b} \xleftarrow{t} \gamma \quad \gamma \xrightarrow{s} o^\nu_{\gamma,a,c} \xleftarrow{t} \gamma, c \quad \gamma \xrightarrow{s} \iota^\nu_{\gamma,a,c} \xleftarrow{t} \gamma, c\,,$$

- plus, for all transition types $\gamma \xrightarrow{s} \alpha \xleftarrow{t} \gamma, \delta$ and bijections $h\colon \gamma \xrightarrow{\sim} \gamma'$ and $k : \delta \xrightarrow{\sim} \delta'$ such that $\gamma' \cap \delta' = \varnothing$, a morphism $(h,k)\colon (h,k) \cdot \alpha \to \alpha$, where

$$(h,k) \cdot \tau_\gamma = \tau_{\gamma'} \qquad \text{and} \qquad (h,k) \cdot \alpha_{\gamma,a,b} = \alpha_{\gamma',h(a),(h+k)(b)} \text{ for } \alpha \in \{o, \iota, o^\nu, \iota^\nu\},$$

satisfying the obvious equations:

$$\begin{array}{ccccc} \gamma & \xrightarrow{\quad s \quad} & \alpha & \xleftarrow{\quad t \quad} & \gamma, \delta \\ {\scriptstyle h}\uparrow & & \uparrow{\scriptstyle (h,k)} & & \uparrow{\scriptstyle h+k} \\ \gamma' & \xrightarrow[\quad s \quad]{} & (h,k) \cdot \alpha & \xleftarrow[\quad t \quad]{} & \gamma', \delta'. \end{array} \tag{11}$$

Notation 5.2. For general presheaves X, we denote the action $X(c') \xrightarrow{X(f)} X(c)$ of any $f : c \to c'$ in the base by $x \mapsto x \cdot f$. However, for $f : \gamma \nrightarrow \gamma'$ in sets, although f acts contravariantly as a map $\gamma' \to \gamma$ in $\mathbb{B}$, it acts covariantly as a set-map, so we often write $f \cdot x$ instead. We use the same convention for arbitrary (potentially non-bijective) maps below.

Proposition 5.3. *The category $\widehat{\mathbb{B}}$ forms a transition category, with transition types $\mathbf{T}^\pi$ as above.*

Let us now define our monad on $\widehat{\mathbb{B}}$, in three stages. We first define $\mathscr{T}_\pi(X)$ on process types. We then observe that not only bijections act on $\mathscr{T}_\pi(X)$, but also arbitrary maps. The presheaf $\mathscr{T}_\pi(X)$ that we defined on process types and bijections thus extends to a functor on the category of finite subsets of $\mathscr{N}$ and all maps. We finally use this to define $\mathscr{T}_\pi(X)$ on transitions.

For any $X \in \widehat{\mathbb{B}}$, let $\mathscr{T}_\pi(X)(\gamma)$ denote the set of all α-equivalence classes of π-calculus terms of the form $\gamma \vdash P$, as defined in the top part of Figure 1. Let us then define *renaming*, the action $\mathscr{T}_\pi(\gamma) \to \mathscr{T}_\pi(\gamma')$ of any set-map $h : \gamma \to \gamma'$, denoted by $P \mapsto h \cdot P$, by straightforward induction, as in the middle left part of Figure 1, where $(b \mapsto c)$ is the unique map $\{b\} \to \{c\}$, and a_γ denotes some globally chosen name not in γ.

Finally, let us define $\mathscr{T}_\pi(X)$ on transitions. For all $\gamma \xrightarrow{s} \alpha \xleftarrow{t} \gamma'$, $\mathscr{T}_\pi(X)(\alpha)$ denotes the set of α-equivalence classes of transitions $P \xrightarrow{\alpha} Q$ with constants in X, for $P \in \mathscr{T}_\pi(X)(\gamma)$ and $Q \in \mathscr{T}_\pi(X)(\gamma')$, as defined in the bottom part of Figure 1, where

- the only binding operations are $in^{a,c}_{b.P}$, $\nu b.R$, and $R \rhd^{\nu b} S$ – binding b;
- a peculiarity is that in $(\!|x|\!)(f)$, the status of names in γ is analogous to that of 'binding' names in contexts, e.g., as b in $a(b).\Box$; in particular they are neither free nor bound, nor α-convertible;
- $[b \mapsto c]$ stands for the map $\gamma, b \to \gamma$ mapping b to c and the rest of γ to itself;
- weakening (i.e., renaming along an injective map) is used implicitly in $L|Q$ and $Q|L$;
- for $\nu b.R$, we define

$$fv(o_{\gamma,a,b}) = fv(\iota_{\gamma,a,b}) = fv(o^\nu_{\gamma,a,b}) = fv(\iota^\nu_{\gamma,a,b}) = \{a, b\} \quad \text{and} \quad fv(\tau_\gamma) = \varnothing$$
$$\text{and} \quad \nu c.\tau_{\gamma,c} = \tau_\gamma \qquad \nu c.\alpha_{(\gamma,c),a,b} = \alpha_{\gamma,a,b}, \text{ for } \alpha \in \{o, \iota, o^\nu, \iota^\nu\},$$

  where $\nu c.\alpha$ is defined iff $c \notin fv(\alpha)$.

This almost defines an assignment $ob(\widehat{\mathbb{B}}) \to ob(\widehat{\mathbb{B}})$: it remains to define the action of maps in $\mathbb{B}$ on $\mathscr{T}_\pi(X)$ and show that it is functorial. For maps of the form $\gamma \to \gamma'$, we merely use renaming; for s and t, we use the sources and targets specified in transitions. For $(h, k) : (h, k) \cdot \alpha \to \alpha$, we define $(h, k) \cdot R$ by induction on R as the middle right part of Figure 1.

Finally, we prove that this is functorial, again by induction. This defines an assignment $ob(\widehat{\mathbb{B}}) \to ob(\widehat{\mathbb{B}})$, which easily extends to a functor $\mathscr{T}_\pi : \widehat{\mathbb{B}} \to \widehat{\mathbb{B}}$: the action of a morphism $F : X \to Y$ in $\widehat{\mathbb{B}}$ is obtained by renaming x, resp. e, according to F in $(\!|x|\!)(f)$, resp. $\langle\!\langle e \rangle\!\rangle(h, k)$.

Remark 5.4 (Replication). The π-calculus standardly features additional operations like guarded sum and replication. Guarded sum may be incorporated readily, but, depending on presentation, replication is less easy. E.g., the compact transition rule $\dfrac{P|!P \xrightarrow{\alpha} Q}{!P \xrightarrow{\alpha} Q}$ does not obviously yield a

$$\frac{x \in X(\gamma) \qquad f \in \mathbf{Set}(\gamma, \gamma')}{\gamma' \vdash_X (\!|x|\!)(f)} \qquad \frac{\gamma \vdash_X P \qquad \gamma \vdash_X Q}{\gamma \vdash_X P|Q} \qquad \frac{}{\gamma \vdash_X 0} \qquad \frac{\gamma, a \vdash_X P}{\gamma \vdash_X \nu a.P}$$

$$\frac{\gamma \vdash_X P \qquad a, b \in \gamma}{\gamma \vdash_X \overline{a}\langle b \rangle.P} \qquad \frac{\gamma, b \vdash_X P \qquad a \in \gamma}{\gamma \vdash_X a(b).P}$$

Process renaming

$$\begin{array}{rcl} h \cdot ((\!|x|\!)(f)) & = & (\!|x|\!)(h \circ f) \\ h \cdot (P|Q) & = & (h \cdot P)|(h \cdot Q) \\ h \cdot 0 & = & 0 \\ h \cdot \nu a_\gamma.P & = & \\ & & \nu a_{\gamma'}.((h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) \\ h \cdot \overline{a}\langle b \rangle.P & = & \overline{h(a)}\langle h(b) \rangle.(h \cdot P) \\ h \cdot a(a_\gamma).P & = & \\ & & h(a)(a_{\gamma'}).((h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) \end{array}$$

Transition renaming

$$\begin{array}{rcl} (h,k) \cdot \langle\!\langle e \rangle\!\rangle (h', k') & = & \langle\!\langle e \rangle\!\rangle (h \circ h', k \circ k') \\ (h, id) \cdot in^{a,c}_{a_{\gamma'}.P} & = & in^{h(a),h(c)}_{a_{\gamma'}.(h + (a_{\gamma'} \mapsto a_{\gamma'})) \cdot P} \\ (h, id) \cdot out^{a,c}_{P} & = & out^{h(a),h(c)}_{h \cdot P} \\ (h, id) \cdot (R \rhd S) & = & ((h, id) \cdot R) \rhd ((h, id) \cdot S) \\ (h, (b \mapsto b')) \cdot in^{a,\nu b}_{P} & = & in^{h(a),\nu b'}_{(h + (b \mapsto b')) \cdot P} \\ (h, (b \mapsto b')) \cdot \nabla b.R & = & \nabla b'.(h + (b \mapsto b'), id) \cdot R \\ (h, id) \cdot (R \rhd^{\nu a_\gamma} S) & = & \\ & & ((h, (a_\gamma \mapsto a_{\gamma'})) \cdot R) \rhd^{\nu a_{\gamma'}} ((h, (a_\gamma \mapsto a_{\gamma'})) \cdot S) \\ (h,k) \cdot \nu a_{\gamma,\delta}.R & = & \nu a_{\gamma,\delta}.(h + (a_{\gamma,\delta} \mapsto a_{\gamma',\delta'}), k) \cdot R \\ (h,k) \cdot (L|Q) & = & ((h,k) \cdot L)|(h \cdot Q) \end{array}$$

+ symmetric cases.

$$\frac{e \in X(\alpha) \qquad \gamma \xrightarrow{s} \alpha \xleftarrow{t} \gamma, \delta \qquad h \colon \gamma \twoheadrightarrow \gamma' \qquad k \colon \delta \twoheadrightarrow \delta' \qquad \gamma' \cap \delta' = \varnothing}{\langle\!\langle e \rangle\!\rangle (h,k) : \gamma' \vdash_X (\!|e \cdot s|\!)(h) \xrightarrow{(h,k) \cdot \alpha} \gamma', \delta' \vdash_X (\!|e \cdot t|\!)(h + k)}$$

$$\frac{\gamma, b \vdash_X P \qquad a, c \in \gamma}{in^{a,c}_{b.P} : \gamma \vdash_X a(b).P \xrightarrow{\iota_{\gamma,a,c}} \gamma \vdash_X [b \mapsto c] \cdot P} \qquad \frac{\gamma \vdash_X P \qquad a, b \in \gamma}{out^{a,b}_{\gamma \vdash_X P} : \gamma \vdash_X \overline{a}\langle b \rangle.P \xrightarrow{o_{\gamma,a,b}} \gamma \vdash_X P}$$

$$\frac{R : \gamma \vdash_X P \xrightarrow{o_{\gamma,a,b}} \gamma \vdash_X P' \qquad S : \gamma \vdash_X Q \xrightarrow{\iota_{\gamma,a,b}} \gamma \vdash_X Q'}{R \rhd S : \gamma \vdash_X (P|Q) \xrightarrow{\tau_\gamma} \gamma \vdash_X (P'|Q')} + \text{sym.} \lhd$$

$$\frac{\gamma, b \vdash_X P \qquad a \in \gamma}{in^{a,\nu b}_{P} : \gamma \vdash_X a(b).P \xrightarrow{\iota^\nu_{\gamma,a,b}} \gamma, b \vdash_X P} \qquad \frac{R : \gamma, b \vdash_X P \xrightarrow{o_{(\gamma,b),a,b}} \gamma, b \vdash_X Q \qquad a \neq b}{\nabla b.R : \gamma \vdash_X \nu b.P \xrightarrow{o^\nu_{\gamma,a,b}} \gamma, b \vdash_X Q}$$

$$\frac{R : \gamma \vdash_X P \xrightarrow{o^\nu_{\gamma,a,b}} \gamma, b \vdash_X P' \qquad S : \gamma \vdash_X Q \xrightarrow{\iota^\nu_{\gamma,a,b}} \gamma, b \vdash_X Q'}{R \rhd^{\nu b} S : \gamma \vdash_X P|Q \xrightarrow{\tau_\gamma} \gamma \vdash_X \nu b.(P'|Q')} + \text{sym.} \lhd^{\nu b}$$

$$\frac{R : \gamma, b \vdash_X P \xrightarrow{\alpha} \gamma, b, \delta \vdash_X Q \qquad b \notin fv(\alpha)}{\nu b.R : \gamma \vdash_X \nu b.P \xrightarrow{\nu b.\alpha} \gamma, \delta \vdash_X \nu b.Q} \qquad \frac{L : \gamma \vdash_X P \xrightarrow{\alpha} \gamma, \delta \vdash_X P' \qquad \gamma \vdash_X Q}{L|Q : \gamma \vdash_X P|Q \xrightarrow{\alpha} \gamma, \delta \vdash_X P'|Q} + Q|L$$

Fig. 1. Syntax, renaming, and labelled transition system for π

familial monad. However, the rules of Sangiorgi and Walker [21, §1.3], a variant of which is reproduced below (without the γ's for readability), do yield a familial monad directly.

$$\frac{P \xrightarrow{\alpha} P'}{!P \xrightarrow{\alpha} P'|!P} \qquad \frac{P \xrightarrow{o_{\gamma,a,b}} P' \qquad P \xrightarrow{\iota_{\gamma,a,b}} P''}{!P \xrightarrow{\tau_\gamma} (P'|P'')|!P} \qquad \frac{P \xrightarrow{o^{\nu}_{\gamma,a,b}} P' \qquad P \xrightarrow{\iota^{\nu}_{\gamma,a,b}} P''}{!P \xrightarrow{\tau_\gamma} \nu b.(P'|P'')|!P}$$

Proposition 5.5. *$\mathscr{T}_\pi$ forms a $\mathbf{T}^\pi_s$-familial monad.*

Proof. To see that $\mathscr{T}_\pi$ is a familial functor, by Lemma 3.8, it is enough to exhibit a functor $E\colon el(\mathscr{T}_\pi(1)) \to \widehat{\mathbb{B}}$ such that $\mathscr{T}_\pi(X)(c) \cong \sum_{R \in \mathscr{T}_\pi(1)(c)} [E(c,R), X]$, naturally in X and $c \in \mathbb{B}$. Elements of $\mathscr{T}_\pi(1)(\gamma)$ are processes of type γ, over exactly one constant process, say $\top_{\gamma'}$ of each type γ'. Elements of $\mathscr{T}_\pi(1)(\alpha)$ are transitions with exactly one constant transition of each type $\gamma \xrightarrow{s} \beta \xleftarrow{t} \gamma'$, say $\top_\beta$, with source $\top_\beta \cdot s = \top_\gamma$ and target $\top_\beta \cdot t = \top_{\gamma'}$.

On objects, we construct E by induction on the depth of the considered derivation (globally for all objects $c \in \mathbb{B}$), as in the top part of Figure 2, relying on some global choice of

- binary coproducts,
- and fresh name a_γ for all $\gamma \in \mathscr{P}_f(\mathscr{N})$, as in the definition of renaming.

The choice of a_γ is in fact completely irrelevant, because for all maps $h : \gamma \twoheadrightarrow \gamma'$, $k\colon \delta \twoheadrightarrow \delta'$ in sets, processes $\gamma \vdash P$, and transitions $R\colon (\gamma \vdash P) \xrightarrow{\alpha} (\gamma, \delta \vdash Q)$, we have by induction:

$$E(h \cdot P) = E(P) \qquad \text{and} \qquad E((h,k) \cdot R) = E(R) \qquad \text{(yes, equality!).} \tag{12}$$

Typically, if $f : \gamma \to \gamma'$, we have by definition $E(f \cdot (\!|\top_\gamma|\!)(id)) = E((\!|\top_\gamma|\!)(f)) = \mathbf{y}_\gamma\colon f$ is not taken into account.

Let us now define E on morphisms:

- For any $f : \gamma \twoheadrightarrow \gamma'$ in sets and $P \in \mathscr{T}_\pi(1)(\gamma)$, we have $f \restriction P\colon (\gamma', f \cdot P) \to (\gamma, P)$ in $el(\mathscr{T}_\pi(1))$ and define $E(f \restriction P)\colon E(\gamma', f \cdot P) \to E(\gamma, P)$ to be just the identity (which makes sense by (12)).
- Similarly, for all $(h,k)\colon (h,k) \cdot \alpha \to \alpha$, let $E((h,k) \restriction R) = id_{E(R)}$.
- Finally, for all $\gamma \xrightarrow{s} \alpha \xleftarrow{t} \gamma, \delta$ and $R \in \mathscr{T}_\pi(1)(\alpha)$, we define $E(s \restriction R)$ and $E(t \restriction R)$ as in the bottom part of Figure 2. Clearly, by induction, $E(s \restriction R)$ lies in ${}^{\boxtimes}((\mathbf{T}^\pi_s)^{\boxtimes})$, so that $\mathscr{T}^\pi$ is $\mathbf{T}^\pi_s$-familial by Remark 3.22.

This shows that $\mathscr{T}_\pi$ is a $\mathbf{T}^\pi_s$-familial functor. Its unit is given by variables, and multiplication is essentially substitution, which is straightforwardly defined by induction in Figure 3. This clearly satisfies naturality in both arguments, as well as associativity and unitality. Finally, cartesianness of η and μ may be checked pointwise (i.e., relative to representable presheaves), and follows by induction. □

As announced and expected, because bisimilarity is not a congruence in π, something must fail:

Proposition 5.6. *The free $\mathscr{T}_\pi$-algebra, $\mathscr{T}_\pi(\varnothing)$, with action given by $\mu_\varnothing$, is not compositional.*

Proof. The problem comes from renaming, as embedded in constant terms $(\!|x|\!)(f)$. Indeed, consider $p = (\overline{a}\langle a\rangle.0|b(c).0)$ (for $a \neq b$), a process involved in a standard counterexample to bisimilarity being a congruence [21]. Letting $f : \{a,b\} \to \{a\}$ denote the unique such map, $(\!|p|\!)(f) \in (\mathscr{T}_\pi)^2(\varnothing)\{a\}$ is mapped by $\mu_{\varnothing,\{a\}}$ to $p' = (\overline{a}\langle a\rangle.0|a(c).0)$. The latter process has a τ-transition to $0|0$, which the former cannot match. □

Definition of E on objects

Processes

$$\begin{array}{rcl} E(\llparenthesis \top_\gamma \rrparenthesis(f)) & = & \mathbf{y}_\gamma \\ E(P|Q) & = & E(P) + E(Q) \\ E(0) & = & \varnothing \\ E(\nu a_\gamma.P) & = & E(P) \\ E(\overline{a}\langle b\rangle.P) & = & E(P) \\ E(a(a_\gamma).P) & = & E(P) \end{array}$$

Transitions

$$\begin{array}{rcl} E(\langle\!\langle \top_\alpha \rangle\!\rangle(h,k)) & = & \mathbf{y}_\alpha \\ E(in^{a,c}_{a_\gamma.P}) & = & E(P) \\ E(out^{a,c}_P) & = & E(P) \\ E(R \rhd S) & = & E(R) + E(S) \\ E(in^{a,\nu b}_P) & = & E(P) \end{array} \qquad \begin{array}{rcl} E(\nabla b.R) & = & E(R) \\ E(R \rhd^{\nu a_\gamma} S) & = & E(R) + E(S) \\ E(\nu a_\gamma.R) & = & E(R) \\ E(L|Q) & = & E(L) + E(Q) \\ E(P|R) & = & E(P) + E(R) \end{array}$$

Definition of E on morphisms: cospans $E(R \cdot s) \xrightarrow{E(s\restriction R)} E(R) \xleftarrow{E(t\restriction R)} E(R \cdot t)$

$$\begin{array}{rl} \langle\!\langle \top_\alpha \rangle\!\rangle(h,k): & \mathbf{y}_\gamma \xrightarrow{\mathbf{y}_s} \mathbf{y}_\alpha \xleftarrow{\mathbf{y}_t} \mathbf{y}_{\gamma,\delta} \\ in^{a,c}_{a_\gamma.P}: & E(a(a_\gamma).P) = E(P) \xrightarrow{id} E(P) \xleftarrow{id} E(P) = E([a_\gamma \mapsto c]\cdot P) \\ out^{a,c}_P: & E(\overline{a}\langle b\rangle.P) = E(P) \xrightarrow{id} E(P) \xleftarrow{id} E(P) \\ R \rhd S: & E(P) + E(Q) \xrightarrow{E(s\restriction R)+E(s\restriction S)} E(R) + E(S) \xleftarrow{E(t\restriction R)+E(t\restriction S)} E(P') + E(Q') \\ in^{a,\nu b}_P: & E(\nu b.P) = E(P) \xrightarrow{id} E(P) \xleftarrow{id} E(P) \\ \nabla b.R: & E(\nu b.P) = E(P) \xrightarrow{E(s\restriction R)} E(R) \xleftarrow{E(t\restriction R)} E(Q) \\ R \rhd^{\nu a_\gamma} S: & E(P) + E(Q) \xrightarrow{E(s\restriction R)+E(s\restriction S)} E(R) + E(S) \xleftarrow{E(t\restriction R)+E(t\restriction S)} E(P') + E(Q') \\ \nu a_\gamma.R: & E(\nu a_\gamma.P) = E(P) \xrightarrow{E(s\restriction R)} E(R) \xleftarrow{E(t\restriction R)} E(Q) = E(\nu a_\gamma.Q) \\ L|Q: & E(P) + E(Q) \xrightarrow{E(s\restriction L)+id} E(L) + E(Q) \xleftarrow{E(t\restriction L)+id} E(P') + E(Q) \\ P|R: & E(P) + E(Q) \xrightarrow{id+E(s\restriction R)} E(P) + E(R) \xleftarrow{id+E(t\restriction R)} E(P) + E(Q') \end{array}$$

Fig. 2. Definition of E

Processes:

$$\begin{array}{rcl} \mu_{X,\gamma}(\llparenthesis p \rrparenthesis(f)) & = & f \cdot p \\ \mu_{X,\gamma}(P|Q) & = & \mu_{X,\gamma}(P)|\mu_{X,\gamma}(Q) \\ \mu_{X,\gamma}(\nu a.P) & = & \nu a.(\mu_X)_{\gamma,a}(P) \\ \mu_{X,\gamma}(0) & = & 0 \\ \mu_{X,\gamma}(\overline{a}\langle b\rangle.P) & = & \overline{a}\langle b\rangle.\mu_{X,\gamma}(P) \\ \mu_{X,\gamma}(a(b).P) & = & a(b).\mu_{X,(\gamma,b)}(P) \end{array}$$

Transitions:

$$\begin{array}{rcl} \mu_{X,\alpha}(\langle\!\langle r \rangle\!\rangle(h,k)) & = & (h,k)\cdot r \\ \mu_{X,\iota_{\gamma,a,c}}(in^{a,c}_{b.P}) & = & in^{a,c}_{b.\mu_{X,(\gamma,b)}(P)} \\ \mu_{X,o_{\gamma,a,b}}(out^{a,b}_P) & = & out^{a,b}_{\mu_{X,\gamma}(P)} \\ \mu_{X,\tau_\gamma}(R \rhd S) & = & \mu_{X,o_{\gamma,a,b}}(R) \rhd \mu_{X,\iota_{\gamma,a,b}}(S) \\ \mu_{X,\iota^\nu_{\gamma,a,b}}(in^{a,\nu b}_P) & = & in^{a,\nu b}_{\mu_{X,(\gamma,b)}(P)} \\ \mu_{X,o^\nu_{\gamma,a,b}}(\nabla b.R) & = & \nabla b.\mu_{X,o_{(\gamma,b),a,b}}(R) \\ \mu_{X,\tau_\gamma}(R \rhd^{\nu b} S) & = & \mu_{X,o^\nu_{\gamma,a,b}}(R) \rhd^{\nu b} \mu_{X,\iota^\nu_{\gamma,a,b}}(S) \\ \mu_{X,\nu c.\alpha}(\nu c.R) & = & \nu c.\mu_{X,\alpha}(R) \\ \mu_{X,\alpha}(L|Q) & = & \mu_{X,\alpha}(L)|\mu_{X,\gamma}(Q) \\ \mu_{X,\alpha}(P|R) & = & \mu_{X,\gamma}(P)|\mu_{X,\alpha}(R). \end{array}$$

Fig. 3. Monad multiplication for $\mathscr{T}_\pi$

Remark 5.7. The root of the problem here is $in^{a,c}_{b.P}$, which forces the syntax for processes to feature renaming, even if only at the level of constants $\llparenthesis x \rrparenthesis(f)$. To emphasise that this does make renaming a proper syntactic operation, it may help to realise that $\mathscr{T}_\pi$ could be defined using an explicit

renaming operation $P[f]$ together with equations describing how it propagates down towards the leaves, e.g., $(P|Q)[f] = P[f]|Q[f]$, to finally integrate with constants: $(\!|x|\!)(g)[f] = (\!|x|\!)(f \circ g)$.

5.2 Non-input congruence

A first, standard way around non-congruence of bisimilarity is to elude the problematic case, and prove that bisimilarity is a congruence for all operators but input. Our framework can cover this by viewing π-calculus as a non-free algebra for a smaller monad $\mathscr{T}_\pi^-$.

Definition 5.8. Let $\mathscr{T}_\pi^-$ denote the sub-functor of $\mathscr{T}_\pi$ obtained by removing $a(b).P$ from the syntax of Figure 1, replacing the rule for $(\!|x|\!)(f)$ by

$$\frac{x \in X(\gamma) \qquad f \in \mathbf{Inj}(\gamma, \gamma')}{\gamma' \vdash_X (\!|x|\!)(f)},$$

where $\mathbf{Inj}(\gamma, \gamma')$ denotes the set of injective maps $\gamma \hookrightarrow \gamma'$, and removing $in_{b.P}^{a,c}$ and $in_P^{a,\nu b}$ from the labelled transition system of Figure 1.

Remark 5.9. We need to retain injective renaming for $L|Q$ and $P|R$ to make sense.

We clearly obtain:

Proposition 5.10. *$\mathscr{T}_\pi^-$ is a $\mathbf{T}_s^\pi$-familial monad.*

But this time, instead of considering $\mathscr{T}_\pi^-(\varnothing)$, which does not even satisfy the input rule, we observe that $\mathscr{T}_\pi(\varnothing)$ forms a $\mathscr{T}_\pi^-$-algebra (because it satisfies all π-calculus rules, hence in particular those of $\mathscr{T}_\pi^-$). We thus obtain:

Proposition 5.11. *The $\mathscr{T}_\pi^-$-algebra $\mathscr{T}_\pi(\varnothing)$ is compositional.*

This gives an alternative proof of Sangiorgi and Walker [21, Theorem 2.2.8(1)], which says that bisimilarity is a *non-input congruence* Sangiorgi and Walker [21, Definition 2.1.23]. In our language:

Corollary 5.12. *Bisimilarity for the $\mathscr{T}_\pi^-$-algebra $\mathscr{T}_\pi(\varnothing)$ is a congruence.*

Proof. By Corollary 3.26 and Propositions 5.10 and 5.11. □

Similarly, we recover Sangiorgi and Walker [21, Lemma 2.3.21]:

Corollary 5.13. *Bisimulation up to non-input context is sound.*

5.3 Wide-open bisimilarity

Another standard solution to the failure of bisimilarity to be a congruence is to resort to wide-open bisimilarity. For this, we need to modify our base category $\mathbb{B}$ to include non-bijective maps $\gamma \to \gamma'$.

Definition 5.14. Let $\mathbb{F}$ be defined just as $\mathbb{B}$ (Definition 5.1), but including all morphisms from $\mathbf{Set}^{op}$ (between relevant objects), instead of just bijections.

Please note that while we include non-bijective morphisms $f : \gamma \to \gamma'$, we do not (need to) do so for transition objects. The presheaf category $\widehat{\mathbb{F}}$ forms a transition category with the same transition types as $\widehat{\mathbb{B}}$, which we denote by $\mathbf{T}^+$.

Let us now adapt our monad on $\widehat{\mathbb{B}}$ to $\widehat{\mathbb{F}}$.

Proposition 5.15. *Replacing the rules for* $(\!|x|\!)(f)$ *and* $\langle\!\langle e \rangle\!\rangle(h,k)$ *in Figure 1 by*

$$\frac{x \in X(\gamma)}{\gamma \vdash_X (\!|x|\!)} \qquad\qquad \frac{e \in X(\alpha)}{\gamma \vdash_X \langle\!\langle e \rangle\!\rangle},$$

and changing the base cases for renaming to $f \cdot (\!|x|\!) = (\!|f \cdot x|\!)$ *and* $(h,k) \cdot \langle\!\langle e \rangle\!\rangle = \langle\!\langle (h,k) \cdot e \rangle\!\rangle$ *yields a functor* $\mathscr{T}_\pi^+ \colon \widehat{\mathbb{F}} \to \widehat{\mathbb{F}}$.

It is not entirely trivial that this again forms a familial monad.

Proposition 5.16. $\mathscr{T}_\pi^+$ *is a* $\mathbf{T}_s^+$*-familial monad.*

Proof. Most of the definitions are adapted from $\mathbb{B}$ to $\mathbb{F}$, modulo the following subtlety. On objects, fresh names a_γ are chosen globally for all γ as in the proof of Proposition 5.5, and only the base cases change to $E(\!|x|\!) = \mathbf{y}_\gamma$ (for $x \in X(\gamma)$) and $E\langle\!\langle e \rangle\!\rangle = \mathbf{y}_\alpha$ (for $e \in X(\alpha)$). However, over $\mathbb{B}$, we had $E(f \cdot P) = E(P)$, which considerably eased the definition of E on morphisms. But because of the new treatment of renaming, this no longer holds over $\mathbb{F}$. E.g., consider $P = (\!|\top_{\{a\}}|\!)$. We have $E(P) = \mathbf{y}_{\{a\}}$, but $E((a \mapsto b) \cdot P) = \mathbf{y}_{\{b\}}$ (where, we recall from (**??**), $(a \mapsto b)$ denotes the unique map $\{a\} \to \{b\}$), so we only get $E(P) \cong E((a \mapsto b) \cdot P)$. The treatment of morphisms thus needs adjustment: for $h : \gamma \nrightarrow \gamma'$ and $k \colon \delta \nrightarrow \delta'$ as in (12), we get inductively-defined, functorial assignments

$$E(h \upharpoonright P) \colon E(h \cdot P) \to E(P) \qquad \text{and} \qquad E((h,k) \upharpoonright R) \colon E((h,k) \cdot R) \to E(R).$$

Finally, the definitions of $E(s \upharpoonright R)$ and $E(t \upharpoonright R)$ also need adjustment for scope-changing constructors $in_P^{a,\nu b}$, $\nabla b.R$, $\nu b.R$, and $R \rhd^{\nu b} S$. E.g., consider the morphism $t \upharpoonright in_{b.P}^{a,c} \colon [b \mapsto c] \cdot P \to in_{b.P}^{a,c}$. We have by definition $E(in_{b.P}^{a,c}) = E((b \mapsto a_\gamma) \cdot P)$ so we define $E(t \upharpoonright in_{b.P}^{a,c})$ to be

$$E([b \mapsto c] \cdot P) \xrightarrow{E([a_\gamma \mapsto c] \upharpoonright (b \mapsto a_\gamma) \cdot P)} E((b \mapsto a_\gamma) \cdot P).$$

This assignment on morphisms may be shown to be functorial by induction. For more detail about the definition and functoriality, see §B.

It is then clear that E makes $\mathscr{T}_\pi^+$ familial. Finally, by definition, morphisms $s \upharpoonright R$ are mapped to coproducts of isomorphisms and maps in $\mathbf{T}_s^+$, up to isomorphism in the arrow category, hence $\mathscr{T}_\pi^+$ is indeed $\mathbf{T}_s^+$-familial. □

The compositionality problem created by $x(\!|f|\!)$ having disappeared, we get:

Proposition 5.17. $\mathscr{T}_\pi^+(\varnothing)$ *is a compositional algebra.*

Remark 5.18. It might be slightly worrying at first to observe that, e.g., $E^+(t \upharpoonright in_{a_{\gamma'}.\top_{\gamma,a_\gamma}}^{a,c})$, namely $\mathbf{y}_\gamma \xrightarrow{\mathbf{y}[a_\gamma \mapsto c]} \mathbf{y}_{\gamma,a_\gamma}$ is definitely not a coproduct of isomorphisms or t-maps. But in fact, any bisimulation is stable under renaming by construction, hence everything works smoothly.

Calling *wide-open* bisimilarity the largest bisimulation relation over $\mathscr{T}_\pi^+(\varnothing)$, we get:

Proposition 5.19. *Wide-open bisimilarity is a congruence.*

Proof. By Corollary 3.27 and Propositions 5.16 and 5.17. □

Similarly, we recover Sangiorgi and Walker [21, Corollary 2.3.25]:

Corollary 5.20. *Wide-open bisimulation up to non-input context is sound.*

Remark 5.21. A different presentation of wide-open bisimilarity may be obtained by augmenting $\mathbb{B}$ with a transition type $\gamma, a \xrightarrow{s} \sigma_{\gamma,a,b} \xleftarrow{t} \gamma$ (for $b \in \gamma$), and adding the rule $P \xrightarrow{[a \mapsto b]} [a \mapsto b] \cdot P$.

Remark 5.22. Similarly, *open* bisimilarity [21, Definition 4.6.2] may be obtained by considering yet another base category, where instead of finite sets of names and (bijective) maps we would have as objects pairs (γ, D) of a finite set γ of names and a *distinction* D, i.e., an irreflexive relation on γ, with maps $(\gamma, D) \to (\gamma', D')$ all maps $f : \gamma \to \gamma'$ *respecting* D, i.e., if $(a, b) \in D$ then $f(a) \neq f(b)$.

6 CONCLUSION AND PERSPECTIVES

We presented a simple abstract framework for studying congruence of bisimilarity, based on familial monads and lifting properties. We then refined the framework to account for soundness of bisimulation up to context, using a weak form of cartesianness instead of lifting properties. We finally showed that the framework flexibly accounts for most known results about congruence of bisimilarity and soundness of bisimulation up to context in the π-calculus. To our knowledge, this is the first abstract framework handling both congruence of bisimilarity and soundness of bisimulation up to context in the presence of binding.

However, although the framework provides abstract, rather general proofs of non-trivial facts, it does not yet come with any format, i.e., means to construct instances from more basic data. An obvious next step is thus to look for such formats, which in our case means automatically constructing familial monads satisfying the relevant hypotheses. An intermediate step would be to show that existing formats give rise to familial monads with compositional initial algebras. Another potential direction is to consider questions related to congruence of bisimilarity and soundness of bisimulation up to context, like Howe's method, environmental bisimulation, or solutions of process equations. We also plan to adapt the framework to weak bisimulation and other kinds of languages, e.g., typed variants of π-calculus, functional languages, or logic programming, which has recently been covered by the bialgebraic approach [2]. Finally, we could consider adapting *pointwise analytic* monads [8] to transition categories. The point would be that they could accomodate *structural congruence*, e.g., a commutative variant of parallel composition, thus providing a new language to study the derivation of labelled transition systems from reduction rules [22].

REFERENCES

[1] Filippo Bonchi, Daniela Petrişan, Damien Pous, and Jurriaan Rot. 2016. A general account of coinduction up-to. *Acta Informatica* (2016), 1–64. https://doi.org/10.1007/s00236-016-0271-4

[2] Filippo Bonchi and Fabio Zanasi. 2015. Bialgebraic Semantics for Logic Programming. *Logical Methods in Computer Science* 11, 1 (2015). https://doi.org/10.2168/LMCS-11(1:14)2015

[3] Gérard Boudol and Ilaria Castellani. 1988. A non-interleaving semantics for CCS based on proved transitions. *Fundamenta Informaticae* XI (1988).

[4] Aurelio Carboni and Peter Johnstone. 1995. Connected limits, familial representability and Artin glueing. *Mathematical Structures in Computer Science* 5, 4 (1995), 441–459. https://doi.org/10.1017/S0960129500001183

[5] Gian Luca Cattani, John Power, and Glynn Winskel. 1998. A Categorical Axiomatics for Bisimulation, See [20], 581–596. https://doi.org/10.1007/BFb0055649

[6] Yves Diers. 1978. Spectres et localisations relatifs à un foncteur. *Comptes rendus hebdomadaires des séances de l'Académie des sciences* 287, 15 (1978), 985–988.

[7] Marcelo P. Fiore and Sam Staton. 2006. A Congruence Rule Format for Name-Passing Process Calculi from Mathematical Structural Operational Semantics. In *Proc. 21st Symposium on Logic in Computer Science* IEEE, 49–58. https://doi.org/10.1109/LICS.2006.7

[8] Richard H. G. Garner and Tom Hirschowitz. 2018. Shapely monads and analytic functors. *Journal of Logic and Computation* 28, 1 (2018), 33–83. https://doi.org/10.1093/logcom/exx029

[9] Mark Hovey. 1999. *Model Categories.* Mathematical Surveys and Monographs, Volume 63, AMS (1999), Vol. 63. American Mathematical Society.

[10] André Joyal, Mogens Nielsen, and Glynn Winskel. 1993. Bisimulation and open maps. In *Proc. 8th Symposium on Logic in Computer Science* IEEE, 418–427. https://doi.org/10.1109/LICS.1993.287566
[11] Bartek Klin. 2011. Bialgebras for structural operational semantics: An introduction. *Theoretical Computer Science* 412, 38 (2011), 5043–5069. https://doi.org/10.1016/j.tcs.2011.03.023
[12] Joachim Kock. 2011. Polynomial functors and trees. *International Mathematics Research Notices* 2011, 3 (2011), 609–673. https://doi.org/10.1093/imrn/rnq068
[13] Saunders Mac Lane. 1998. *Categories for the Working Mathematician* (2nd ed.). Number 5 in Graduate Texts in Mathematics. Springer.
[14] Saunders Mac Lane and Ieke Moerdijk. 1992. *Sheaves in Geometry and Logic: A First Introduction to Topos Theory.* Springer.
[15] MohammadReza Mousavi, Michel A. Reniers, and Jan Friso Groote. 2007. SOS Formats and Meta-Theory: 20 Years After. *Theoretical Computer Science* 373, 3 (2007), 238–272.
[16] Gordon D. Plotkin. 1981. *A Structural Approach to Operational Semantics.* DAIMI Report FN-19. Computer Science Department, Aarhus University.
[17] Damien Pous and Davide Sangiorgi. 2011. *Enhancements of the bisimulation proof method.* Number 52 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, Chapter 6.
[18] Emily Riehl. 2014. *Categorical Homotopy Theory.* Number 24 in New Mathematical Monographs. Cambridge University Press.
[19] Davide Sangiorgi. 1996. A theory of bisimulation for the π-calculus. *Acta Informatica* 33, 1 (1996), 69–97. https://doi.org/10.1007/s002360050036
[20] Davide Sangiorgi and Robert de Simone (Eds.). 1998. *Proc. 9th International Conference on Concurrency Theory.* LNCS, Vol. 1466. Springer.
[21] Davide Sangiorgi and David Walker. 2001. *The π-calculus – A Theory of Mobile Processes.* Cambridge University Press.
[22] Peter Sewell. 1998. From Rewrite Rules to Bisimulation Congruences, See [20], 269–284.
[23] Sam Staton. 2008. General Structural Operational Semantics through Categorical Logic. In *Proc. 23rd Symposium on Logic in Computer Science* 166–177. https://doi.org/10.1109/LICS.2008.43
[24] Daniele Turi and Gordon D. Plotkin. 1997. Towards a Mathematical Operational Semantics. In *Proc. 12th Symposium on Logic in Computer Science* 280–291. https://doi.org/10.1109/LICS.1997.614955
[25] Mark Weber. 2007. Familial 2-functors and parametric right adjoints. *Theory and Applications of Categories* 18, 22 (2007), 665–732.

A PROOFS FOR $\mathscr{T}_{CL}$

Proof of Proposition 3.2. Compositionality means that for any transition $r\colon \mu_{\varnothing,\star}(M) \to Y$, there exists N and

$$R \in \mathscr{T}_{CL}(\mathscr{T}_{CL}(\varnothing))(M, N)$$

such that $r = \mu_{\varnothing,[1]}(R)$ (and $Y = \mu_{\varnothing,\star}(N)$). Here, M is merely a term with variables in $\mathscr{T}_{CL}(\varnothing)$, otherwise said a pair of a context C with a certain number of free variables occurrences (ordered from left to right), say $x_1, \ldots, x_n$, together with n terms $M_1, \ldots, M_n$. We denote this by $M = C[M_1, \ldots, M_n]$. We proceed by induction on C and case analysis on r:

- If $C = (\!|x_1|\!)$ (and $n = 1$), then we have
$$\mu_{\varnothing,\star}(M) = \mu_{\varnothing,\star}(C[M_1]) = \mu_{\varnothing,\star}(\!|M_1|\!) = M_1.$$
Thus, r is in fact a transition $M_1 \to Y$, so that taking $N = (\!|Y|\!)$ and $R = (\!|r|\!)$, we get $\mu_{\varnothing,[1]}(R) = r$ as desired.
- If $C = C_1\ C_2$, then we may divide $M_1, \ldots, M_n$ into
$$M_1^1, \ldots, M_{n_1}^1, M_1^2, \ldots, M_{n_2}^2,$$
so that putting $X = \mu_{\varnothing,\star}(M)$, $P_i = C_i[M_1^i, \ldots, M_{n_i}^i]$, and $X_i = \mu_{\varnothing,\star}(P_i)$, we have $M = P_1\ P_2$ and hence $X = X_1\ X_2$. We then proceed by case analysis.
  - If $r = r_1\ X_2$, for some $r_1\colon X_1 \to Y_1$, then $Y = Y_1\ X_2$, and by induction hypothesis we get $R_1\colon P_1 \to Q_1$ such that $\mu_{\varnothing,[1]}(R_1) = r_1$ and $\mu_{\varnothing,\star}(Q_1) = Y_1$. Taking $R = R_1\ P_2$ thus yields $\mu_{\varnothing,[1]}(R) = r_1\ X_2$ as desired.

– The case $r = X_1\ r_2$ is similar.
– If $X_1 = K\ X'$ and $r = k_{X',X_2}$, then $C_1 = K\ C'$ and, putting $P' = C'[M_1^1, ..., M_{n_1}^1]$, we have $X' = \mu_{\varnothing,\star}(P')$ and $Y = X'$. Taking $R = k_{P',P_2}$ then yields $\mu_{\varnothing,[1]}(R) = k_{X',X_2}$ as desired.
– The case where $X_1 = S\ X'\ X''$ and $r = s_{X',X'',X_2}$ is similar. □

Proof of Proposition 3.9. By Lemma 3.8, it suffices to exhibit a functor $E\colon el(\mathscr{T}_{CL}(1)) \to \mathbf{Gph}$, such that

$$\mathscr{T}_{CL}(Z)(c) \cong \sum_{x \in \mathscr{T}_{CL}(1)(c)} [E(c,x), Z],$$

naturally in c and Z. Now, $\mathscr{T}_{CL}(1)(\star)$ consists of closed terms on a unique free variable, say $\top$, and we define E to map any such term C to the discrete graph with vertices in the ordinal n_C, where n_C is the number of occurrences of $\top$ in C. Similarly, $\mathscr{T}_{CL}[1]$ consists of transition derivations on just one transition axiom, say $\pi\colon \top \to \top$. On such derivations, we define E by induction:

$$\begin{array}{rclcrcl} E(\!|\pi|\!) & = & \mathbf{y}_{[1]} & & E(L\ N) & = & E(L) + E(N) \\ E(s_{M,N,P}) & = & E(M) + E(N) + E(P) & & E(M\ R) & = & E(M) + E(R). \\ E(k_{M,N}) & = & E(M) + E(N) & & & & \end{array}$$

Finally, we need to define E on s and t. We again proceed inductively:

- For the variable case, we have $E(\!|\top|\!) = \star$ and $E(\!|\pi|\!) = [1]$, so we let $E(s \upharpoonright \pi)$ be just s and $E(t \upharpoonright \pi)$ be t.
- For $s_{M,N,P}$, we have $E(M\ N\ P) \cong E(M) + E(N) + E(P)$ and $E((M\ N)\ (M\ P)) = E(M) + E(N) + E(M) + E(P)$, so we pick the obvious maps

$$E(M) + E(N) + E(P) \quad \to \quad E(M) + E(N) + E(P) \quad \leftarrow \quad E(M) + E(N) + E(M) + E(P)$$

  for $E(s \upharpoonright s_{M,N,P})$ and $E(t \upharpoonright s_{M,N,P})$.
- We proceed similarly for $k_{M,N}$.
- For $L\ N$, we get by induction hypothesis $E(M) \xrightarrow{E(s\upharpoonright L)} E(L) \xleftarrow{E(t\upharpoonright L)} E(M')$, which induces

$$E(M) + E(N) \xrightarrow{E(s\upharpoonright L)+E(N)} E(L) + E(N) \xleftarrow{E(t\upharpoonright L)+E(N)} E(M') + E(N).$$

- We proceed similarly for $M\ R$. □

B DEFINITION OF E ON MORPHISMS, AND FUNCTORIALITY

In this section, we give the full definition of $E\colon el(\mathscr{T}_\pi^+(1)) \to \widehat{\mathbb{F}}$, the functor used to show that $\mathscr{T}_\pi^+$ is familial (the definition is given on objects in the body, but the definition on morphisms is a bit elliptic), and prove its functoriality.

Before starting, let us recall that for all $a \in \gamma$ and $b, c \notin \gamma$, $[b \mapsto a]$ denotes the obvious map $\gamma, b \to \gamma$, while $(b \mapsto c)$ denotes the isomorphism $\gamma, b \to \gamma, c$ – γ is implicit in the notation but hopefully may be inferred from context.

B.1 Definition of E on renaming

The definition is in Figure 4. On processes, this includes non-injective renaming. The pattern is: $E(f \upharpoonright X)\colon E(f \cdot X) \to E(X)$.

Lemma B.1 (Functoriality of E on renaming). *For all $\gamma \xrightarrow{h} \gamma' \xrightarrow{h'} \gamma''$ and $\gamma \vdash_1 P$, we have*

$$E(h' \circ h \upharpoonright P) = E(h \upharpoonright P) \circ E(h' \upharpoonright h \cdot P).$$

$E(h \restriction (\!|\top_\gamma|\!))\colon E(h \cdot (\!|\top_\gamma|\!)) = E(\!|\top_{\gamma'}|\!) = \mathbf{y}_{\gamma'} \xrightarrow{\mathbf{y}_h} \mathbf{y}_\gamma = E(\!|\top_\gamma|\!)$

$E(h \restriction (P|Q))\colon E(h \cdot (P|Q)) = E(h \cdot P) + E(h \cdot Q) \xrightarrow{E(h\restriction P)+E(h\restriction Q)} E(P) + E(Q)$

$E(h \restriction 0)\colon E(h \cdot 0) = 0 \xrightarrow{id} 0$

$E(h \restriction \nu a_\gamma.P)\colon E(h \cdot \nu a_\gamma.P) = E((h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) \xrightarrow{E((h+(a_\gamma\mapsto a_{\gamma'}))\restriction P)} E(P)$

$E(h \restriction \overline{a}\langle b\rangle.P)\colon E(h \cdot \overline{a}\langle b\rangle.P) = E(h \cdot P) \xrightarrow{E(h\restriction P)} E(P)$

$E(h \restriction a(a_\gamma).P)\colon E(h \cdot a(a_\gamma).P) = E((h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) \xrightarrow{E(h+(a_\gamma\mapsto a_{\gamma'})\restriction P)} E(P)$

$E((h,k) \restriction \langle\!\langle \top_\alpha \rangle\!\rangle)\colon E\langle\!\langle \top_{(h,k)\cdot\alpha} \rangle\!\rangle = \mathbf{y}_{(h,k)\cdot\alpha} \xrightarrow{\mathbf{y}_{(h,k)}} \mathbf{y}_\alpha$

$E((h,id) \restriction in^{a,c}_{a_\gamma.P})\colon E((h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) \xrightarrow{E(h+(a_\gamma\mapsto a_{\gamma'})\restriction P)} E(P)$

$E((h,id) \restriction out^{a,c}_P)\colon E((h,id) \cdot out^{a,c}_P) = E(h \cdot P) \xrightarrow{E(h\restriction P)} E(P)$

$E((h,id) \restriction R \rhd S)\colon E((h,id) \cdot R) + E((h,id) \cdot S) \xrightarrow{E((h,id)\restriction R)+E((h,id)\restriction S)} E(R) + E(S)$

$E((h,(b \mapsto b')) \restriction in^{a,\nu b}_P)\colon E((h + (b \mapsto b')) \cdot P) \xrightarrow{E(h+(b\mapsto b')\restriction P)} E(P)$

$E((h,(b \mapsto b')) \restriction \nabla b.R)\colon E((h + (b \mapsto b'), id) \cdot R) \xrightarrow{E((h+(b\mapsto b'),id)\restriction R)} E(R)$

$E((h,id) \restriction R_1 \rhd^{\nu a_\gamma} R_2) : \sum_i E((h,(a_\gamma \mapsto a_{\gamma'})) \cdot R_i) \xrightarrow{\sum_i E((h,(a_\gamma\mapsto a_{\gamma'}))\restriction R_i)} \sum_i E(R_i)$

$E((h,k) \restriction \nu a_{\gamma,\delta}.R) : E((h + (a_{\gamma,\delta} \mapsto a_{\gamma',\delta'}), k) \cdot R) \xrightarrow{E((h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}),k)\restriction R)} E(R)$

$E((h,k) \restriction (L|Q))\colon E((h,k) \cdot L) + E(h \cdot Q) \xrightarrow{E((h,k)\restriction L)+E(h\restriction Q)} E(L) + E(Q)$

+ symmetric cases.

Fig. 4. Definition of E on renaming

For all $R\colon (\gamma \vdash_1 P) \xrightarrow{\alpha} (\gamma, \delta \vdash_1 Q)$, $\gamma \xrightarrow{h} \gamma' \xrightarrow{h'} \gamma''$, *and* $\delta \xrightarrow{k} \delta' \xrightarrow{k'} \delta''$, *assuming* $\gamma' \cap \delta' = \gamma'' \cap \delta'' = \varnothing$, *we have*

$$E((h',k') \circ (h,k) \restriction R) = E((h,k) \restriction R) \circ E((h',k') \restriction (h,k) \cdot R).$$

Proof. We prove the first point by induction on P, and the second by induction on R, using the former.

Case $(\!|\top_\gamma|\!)$**.**

$$E(h \restriction (\!|\top_\gamma|\!)) \circ E(h' \restriction h \cdot (\!|\top_\gamma|\!)) = \mathbf{y}_h \circ \mathbf{y}_{h'} = \mathbf{y}_{h' \circ h} = E(h' \circ h \restriction (\!|\top_\gamma|\!)),$$

where of course $h' \circ h$ is taken in sets, not in $\mathbb{F}$.

Case $P|Q$**.**

$$\begin{aligned}
& E(h \restriction P|Q) \circ E(h' \restriction h \cdot (P|Q)) \\
= \;& (E(h \restriction P) + E(h \restriction Q)) \circ (E(h' \restriction h \cdot P) + E(h' \restriction h \cdot Q)) && \text{(by definition)} \\
= \;& (E(h \restriction P) \circ E(h' \restriction h \cdot P)) + (E(h \restriction Q) \circ E(h' \restriction h \cdot Q)) && \text{(by interchange)} \\
= \;& E(h' \circ h \restriction P) + E(h' \circ h \restriction Q) && \text{(by induction hypothesis)} \\
= \;& E(h' \circ h \restriction P|Q) && \text{(by definition).}
\end{aligned}$$

Case 0. Trivial.
Case $va_\gamma.P$.

$$\begin{aligned}
& E(h \restriction va_\gamma.P) \circ E(h' \restriction h \cdot va_\gamma.P) \\
= \;& E(h + (a_\gamma \mapsto a_{\gamma'}) \restriction P) \circ E(h' + (a_{\gamma'} \mapsto a_{\gamma''}) \restriction (h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) && \text{(by definition)} \\
= \;& E((h' \circ h) + (a_\gamma \mapsto a_{\gamma''}) \restriction P) && \text{(by induction hypothesis)} \\
= \;& E(h' \circ h \restriction va_\gamma.P) && \text{(by definition).}
\end{aligned}$$

Case $\overline{a}\langle b\rangle.P$.

$$\begin{aligned}
& E(h \restriction \overline{a}\langle b\rangle.P) \circ E(h' \restriction h \cdot \overline{a}\langle b\rangle.P) \\
= \;& E(h \restriction P) \circ E(h' \restriction h \cdot P) && \text{(by definition)} \\
= \;& E(h' \circ h \restriction P) && \text{(by induction hypothesis)} \\
= \;& E(h' \circ h \restriction \overline{a}\langle b\rangle.P) && \text{(by definition).}
\end{aligned}$$

Case $a(a_\gamma).P$.

$$\begin{aligned}
& E(h \restriction a(a_\gamma).P) \circ E(h' \restriction h \cdot a(a_\gamma).P) \\
= \;& E(h + (a_\gamma \mapsto a_{\gamma'}) \restriction P) \circ E(h' + (a_{\gamma'} \mapsto a_{\gamma''}) \restriction (h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) && \text{(by definition)} \\
= \;& E((h' \circ h) + (a_\gamma \mapsto a_{\gamma''}) \restriction P) && \text{(by induction hypothesis)} \\
= \;& E(h' \circ h \restriction a(a_\gamma).P) && \text{(by definition).}
\end{aligned}$$

Let us now consider transitions.
Case $\langle\!\langle \top_\alpha \rangle\!\rangle$.

$$\begin{aligned}
& E((h,k) \restriction \langle\!\langle \top_\alpha \rangle\!\rangle) \circ E((h',k') \restriction (h,k) \cdot \langle\!\langle \top_\alpha \rangle\!\rangle) \\
= \;& \mathbf{y}_{(h,k)} \circ \mathbf{y}_{(h',k')} \\
= \;& \mathbf{y}_{(h' \circ h, k' \circ k)} \\
= \;& E((h' \circ h, k' \circ k) \restriction \langle\!\langle \top_\alpha \rangle\!\rangle).
\end{aligned}$$

Case $in^{a,c}_{a_\gamma.P}$.

$$\begin{aligned}
& E((h,id) \restriction in^{a,c}_{a_\gamma.P}) \circ E((h',id) \restriction (h,id) \cdot in^{a,c}_{a_\gamma.P}) \\
= \;& E(h + (a_\gamma \mapsto a_{\gamma'}) \restriction P) \circ E(h' + (a_{\gamma'} \mapsto a_{\gamma''}) \restriction h + (a_\gamma \mapsto a_{\gamma'}) \cdot P) \\
= \;& E((h' \circ h) + (a_\gamma \mapsto a_{\gamma''}) \restriction P) && \text{(by functoriality on processes)} \\
= \;& E((h' \circ h, id) \restriction in^{a,c}_{a_\gamma.P}).
\end{aligned}$$

Case $out^{a,b}_P$.

$$\begin{aligned}
& E((h,id) \restriction out^{a,b}_P) \circ E((h',id) \restriction (h,id) \cdot out^{a,b}_P) \\
= \;& E(h \restriction P) \circ E(h' \restriction h \cdot P) \\
= \;& E(h' \circ h \restriction P) && \text{(by functoriality on processes)} \\
= \;& E((h' \circ h, id) \restriction out^{a,b}_P).
\end{aligned}$$

Case $R \rhd S$.

$$\begin{aligned}
& E((h,id) \restriction R \rhd S) \circ E((h',id) \restriction (h,id) \cdot R \rhd S) \\
= \;& (E((h,id) \restriction R) \circ E((h',id) \restriction (h,id) \cdot R)) + (E((h,id) \restriction S) \circ E((h',id) \restriction (h,id) \cdot S)) \\
= \;& E((h' \circ h, id) \restriction R) + E((h' \circ h, id) \restriction S) && \text{(by induction hypothesis)} \\
= \;& E((h' \circ h, id) \restriction R \rhd S).
\end{aligned}$$

Case $in_P^{a,\nu b}$.

$$\begin{aligned}
& E((h,(b\mapsto b'))\restriction in_P^{a,\nu b})\circ E((h',(b'\mapsto b''))\restriction(h,(b\mapsto b'))\cdot in_P^{a,\nu b}) \\
= \;& E(h+(b\mapsto b')\restriction P)\circ E(h'+(b'\mapsto b'')\restriction h+(b\mapsto b')\cdot P) \\
= \;& E((h'\circ h)+(b\mapsto b'')\restriction P) \qquad \text{(by functoriality on processes)} \\
= \;& E((h'\circ h,(b\mapsto b''))\restriction in_P^{a,\nu b}).
\end{aligned}$$

Case $\nabla b.R$.

$$\begin{aligned}
& E((h,(b\mapsto b'))\restriction \nabla b.R)\circ E((h',(b'\mapsto b''))\restriction(h,(b\mapsto b'))\cdot \nabla b.R) \\
= \;& E((h+(b\mapsto b'),id)\restriction R)\circ E((h'+(b'\mapsto b''),id)\restriction(h+(b\mapsto b'),id)\cdot R) \\
= \;& E(((h'\circ h)+(b\mapsto b''),id)\restriction R) \qquad \text{(by induction hypothesis)} \\
= \;& E((h'\circ h,(b\mapsto b''))\restriction \nabla b.R).
\end{aligned}$$

Case $R \rhd^{\nu a_\gamma} S$.

$$\begin{aligned}
& E((h,id)\restriction R\rhd^{\nu a_\gamma} S)\circ E((h',id)\restriction(h,id)\cdot R\rhd^{\nu a_\gamma} S) \\
= \;& (E((h,(a_\gamma\mapsto a_{\gamma'}))\restriction R)\circ E((h',(a_{\gamma'}\mapsto a_{\gamma''}))\restriction(h,(a_\gamma\mapsto a_{\gamma'}))\cdot R)) \\
& +(E((h,(a_\gamma\mapsto a_{\gamma'}))\restriction S)\circ E((h',(a_{\gamma'}\mapsto a_{\gamma''}))\restriction(h,(a_\gamma\mapsto a_{\gamma'}))\cdot S)) \\
= \;& E((h'\circ h,(a_\gamma\mapsto a_{\gamma''}))\restriction R)+E((h'\circ h,(a_\gamma\mapsto a_{\gamma''}))\restriction S) \;\text{(by induction hypothesis)} \\
= \;& E((h'\circ h,id)\restriction R\rhd^{\nu a_\gamma} S).
\end{aligned}$$

Case $\nu a_{\gamma,\delta}.R$.

$$\begin{aligned}
& E((h,k)\restriction \nu a_{\gamma,\delta}.R)\circ E((h',k')\restriction(h,k)\cdot \nu a_{\gamma,\delta}.R) \\
= \;& (E((h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}),k)\restriction R)\circ E((h'+(a_{\gamma',\delta'}\mapsto a_{\gamma'',\delta''}),k')\restriction(h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}),k)\cdot R)) \\
= \;& E(((h'\circ h)+(a_{\gamma,\delta}\mapsto a_{\gamma'',\delta''}),k'\circ k)\restriction R) \;\text{(by induction hypothesis)} \\
= \;& E((h'\circ h,k'\circ k)\restriction \nu a_{\gamma,\delta}.R).
\end{aligned}$$

Case $L|Q$.

$$\begin{aligned}
& E((h,k)\restriction L|Q)\circ E((h',k')\restriction(h,k)\cdot(L|Q)) \\
= \;& (E((h,k)\restriction L)+E(h\restriction Q))\circ(E((h',k')\restriction(h,k)\cdot L)+E(h'\restriction h\cdot Q)) \\
= \;& (E((h,k)\restriction L)\circ E((h',k')\restriction(h,k)\cdot L))+(E(h\restriction Q)+E(h'\restriction h\cdot Q)) \\
= \;& E((h'\circ h,k'\circ k)\restriction L)+E(h'\circ h\restriction Q) \\
& \qquad \text{by induction hypothesis and functoriality on processes} \\
= \;& E((h'\circ h,k'\circ k)\restriction L|Q).
\end{aligned}$$

Cases $P|R$, $S\lhd R$, $S\lhd^{\nu a_\gamma} R$.. Symmetric. □

B.2 Definition of E on morphisms: cospans $E(R\cdot s)\xrightarrow{E(s\restriction R)} E(R)\xleftarrow{E(t\restriction R)} E(R\cdot t)$

We assume that constructors are as in the figure introducing the labelled transition system, except when visible otherwise. E.g., when we write $R\rhd S$, it is implicitly assumed that $R\colon P\xrightarrow{o_{\gamma,a,b}} P'$ and $S\colon Q\xrightarrow{\iota_{\gamma,a,b}} Q'$. By contrast, when we write $in^{a,c}_{a_{\gamma'}.P}$, we implicitly assume that the necessary renamings have been performed.

Furthermore, $i^{\delta}_{\gamma}\colon \gamma\hookrightarrow\gamma,\delta$ denotes the inclusion.

Let us now define E on $s\restriction R$ and $t\restriction R$ by induction on R:

$\langle\!\langle \top_\alpha \rangle\!\rangle$: $\quad \mathbf{y}_\gamma \xrightarrow{\mathbf{y}_s} \mathbf{y}_\alpha \xleftarrow{\mathbf{y}_t} \mathbf{y}_{\gamma,\delta} \qquad (\text{if } \gamma \xrightarrow{s} \alpha \xleftarrow{t} \gamma, \delta)$

$in^{a,c}_{a_\gamma.P}$: $\quad E(P) \xrightarrow{id} E(P) \xleftarrow{E([a_\gamma \mapsto c] \restriction P)} E([a_\gamma \mapsto c] \cdot P)$

$out^{a,c}_P$: $\quad E(P) \xrightarrow{id} E(P) \xleftarrow{id} E(P)$

$R \rhd S$: $\quad E(P) + E(Q) \xrightarrow{E(s\restriction R)+E(s\restriction S)} E(R) + E(S) \xleftarrow{E(t\restriction R)+E(t\restriction S)} E(P') + E(Q')$

$in^{a,\nu b}_P$: $\quad E((b \mapsto a_\gamma) \cdot P) \xrightarrow{E((b\mapsto a_\gamma)\restriction P)} E(P) \xleftarrow{id} E(P)$

$\nabla b.R$: $\quad E((b \mapsto a_\gamma) \cdot P) \xrightarrow{E((b\mapsto a_\gamma)\restriction P)} E(P) \xrightarrow{E(s\restriction R)} E(R) \xleftarrow{E(t\restriction R)} E(Q)$

$R \rhd^{\nu a_\gamma} S$: $\quad E(P) + E(Q) \xrightarrow{E(s\restriction R)+E(s\restriction S)} E(R) + E(S) \xleftarrow{E(t\restriction R)+E(t\restriction S)} E(P') + E(Q')$

$\nu a_{\gamma,\delta}.R$: $\quad E((a_{\gamma,\delta} \mapsto a_\gamma) \cdot P) \xrightarrow{E((a_{\gamma,\delta}\mapsto a_\gamma)\restriction P)} E(P) \xrightarrow{E(s\restriction R)} E(R) \xleftarrow{E(t\restriction R)} E(Q)$

$L|Q$: $\quad E(P) + E(Q) \xrightarrow{E(s\restriction L)+id} E(L) + E(Q) \xleftarrow{E(t\restriction L)+E(i^\delta_\gamma\restriction Q)} E(P') + E(i^\delta_\gamma \cdot Q)$

+ symmetric cases.

B.3 Functoriality

We have already seen that E is functorial on morphisms of the form $h \restriction P$ and $(h,k) \restriction R$, for appropriate h and k. The only missing piece to its functoriality is thus:

Lemma B.2 (Functoriality for the defining equations of $\mathbb{F}$). *For all R: $(\gamma \vdash_1 P) \xrightarrow{\alpha} (\gamma, \delta \vdash_1 Q)$, $\gamma \xrightarrow{h} \gamma'$, and $\delta \xrightarrow{k} \delta'$ such that $\gamma' \cap \delta' = \varnothing$, we have $E((h,k) \restriction R) \circ E(s \restriction (h,k) \cdot R) = E(s \restriction R) \circ E(h \restriction P)$ and $E((h,k) \restriction R) \circ E(t \restriction (h,k) \cdot R) = E(t \restriction R) \circ E(h + k \restriction Q)$.*

Proof. We again proceed by induction, often abbreviating $(h, id) \cdot R$ to $h \cdot R$.

Case $\langle\!\langle \top_\alpha \rangle\!\rangle$. The squares

$$\begin{array}{ccc}
E\langle\!\langle \top_{\gamma'} \rangle\!\rangle & \xrightarrow{E(h\restriction\langle\!\langle \top_\gamma \rangle\!\rangle)} & E\langle\!\langle \top_\gamma \rangle\!\rangle \\
{\scriptstyle E(s\restriction\langle\!\langle \top_{(h,k)\cdot\alpha} \rangle\!\rangle)}\downarrow & & \downarrow{\scriptstyle E(s\restriction\langle\!\langle \top_\alpha \rangle\!\rangle)} \\
E\langle\!\langle \top_{(h,k)\cdot\alpha} \rangle\!\rangle & \xrightarrow[E((h,k)\restriction\langle\!\langle \top_\alpha \rangle\!\rangle)]{} & E\langle\!\langle \top_\alpha \rangle\!\rangle \\
{\scriptstyle E(t\restriction\langle\!\langle \top_{(h,k)\cdot\alpha} \rangle\!\rangle)}\uparrow & & \uparrow{\scriptstyle E(t\restriction\langle\!\langle \top_\alpha \rangle\!\rangle)} \\
E\langle\!\langle \top_{\gamma',\delta'} \rangle\!\rangle & \xrightarrow[E(h+k\restriction\langle\!\langle \top_{\gamma,\delta} \rangle\!\rangle)]{} & E\langle\!\langle \top_{\gamma,\delta} \rangle\!\rangle
\end{array}$$

unfold to

$$\begin{array}{ccc}
\mathbf{y}_{\gamma'} & \xrightarrow{\mathbf{y}_h} & \mathbf{y}_\gamma \\
{\scriptstyle \mathbf{y}_s}\downarrow & & \downarrow{\scriptstyle \mathbf{y}_s} \\
\mathbf{y}_{(h,k)\cdot\alpha} & \xrightarrow{\mathbf{y}_{(h,k)}} & \mathbf{y}_\alpha \\
{\scriptstyle \mathbf{y}_t}\uparrow & & \uparrow{\scriptstyle \mathbf{y}_t} \\
\mathbf{y}_{\gamma',\delta'} & \xrightarrow[\mathbf{y}_{(h+k)}]{} & \mathbf{y}_{\gamma,\delta}.
\end{array}$$

Case $in^{a,c}_{a_\gamma.P}$.

$$\begin{array}{ccc}
E(h \cdot a(a_\gamma).P) & \xrightarrow{E(h \restriction a(a_\gamma).P)} & E(a(a_\gamma).P) \\
{\scriptstyle E(s \restriction h \cdot in^{a,c}_{a_\gamma.P})}\downarrow & & \downarrow{\scriptstyle E(s \restriction in^{a,c}_{a_\gamma.P})} \\
E(h \cdot in^{a,c}_{a_\gamma.P}) & \xrightarrow{E(h \restriction in^{a,c}_{a_\gamma.P})} & E(in^{a,c}_{a_\gamma.P}) \\
{\scriptstyle E(t \restriction h \cdot in^{a,c}_{a_\gamma.P})}\uparrow & & \uparrow{\scriptstyle E(t \restriction in^{a,c}_{a_\gamma.P})} \\
E([h, a_\gamma \mapsto h(c)] \cdot P) & \xrightarrow[E(h \restriction [a_\gamma \mapsto c] \cdot P)]{} & E([a_\gamma \mapsto c] \cdot P)
\end{array}$$

unfolds to

$$\begin{array}{ccc}
E((h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) & \xrightarrow{E(h + (a_\gamma \mapsto a_{\gamma'}) \restriction P)} & E(P) \\
\| & & \| \\
E((h + (a_\gamma \mapsto a_{\gamma'})) \cdot P) & \xrightarrow{E(h + (a_\gamma \mapsto a_{\gamma'}) \restriction P)} & E(P) \\
{\scriptstyle E([a_{\gamma'} \mapsto h(c)] \restriction P')}\uparrow & & \uparrow{\scriptstyle E([a_{\gamma'} \mapsto c] \restriction P)} \\
E([h, a_\gamma \mapsto h(c)] \cdot P) & \xrightarrow[E(h \restriction [a_\gamma \mapsto c] \cdot P)]{} & E([a_\gamma \mapsto c] \cdot P)),
\end{array}$$

where $P' = (h + (a_\gamma \mapsto a_{\gamma'})) \cdot P$, which commutes by Lemma B.1.

Case $out^{a,b}_P$**.**

$$\begin{array}{ccc}
E(h \cdot \overline{a}\langle b \rangle.P) & \xrightarrow{E(h \restriction \overline{a}\langle b \rangle.P)} & E(\overline{a}\langle b \rangle.P) \\
{\scriptstyle E(s \restriction h \cdot out^{a,b}_P)}\downarrow & & \downarrow{\scriptstyle E(s \restriction out^{a,b}_P)} \\
E(h \cdot out^{a,b}_P) & \xrightarrow{E(h \restriction out^{a,b}_P)} & E(out^{a,b}_P) \\
{\scriptstyle E(t \restriction h \cdot out^{a,b}_P)}\uparrow & & \uparrow{\scriptstyle E(t \restriction out^{a,b}_P)} \\
E(h \cdot P) & \xrightarrow[E(h \restriction P)]{} & E(P)
\end{array}$$

unfolds to

$$\begin{array}{ccc}
E(h \cdot P) & \xrightarrow{E(h \restriction P)} & E(P) \\
\| & & \| \\
E(h \cdot P) & \xrightarrow{E(h \restriction P)} & E(P) \\
\| & & \| \\
E(h \cdot P) & \xrightarrow[E(h \restriction P)]{} & E(P),
\end{array}$$

Case $R \rhd S$**.**

$$\begin{array}{ccc}
E(h \cdot (P|Q)) & \xrightarrow{E(h \restriction (P|Q))} & E(P|Q) \\
{\scriptstyle E(s \restriction (h,id) \cdot (R \rhd S))}\downarrow & & \downarrow{\scriptstyle E(s \restriction R \rhd S)} \\
E((h, id) \cdot (R \rhd S)) & \xrightarrow{E((h,id) \restriction R \rhd S)} & E(R \rhd S) \\
{\scriptstyle E(t \restriction (h,id) \cdot R \rhd S)}\uparrow & & \uparrow{\scriptstyle E(t \restriction R \rhd S)} \\
E(h \cdot (P'|Q')) & \xrightarrow[E(h \restriction \nu a_{\gamma,\delta} \cdot (P'|Q'))]{} & E(P'|Q')
\end{array}$$

unfolds to

$$\begin{array}{ccc}
E(h\cdot P)+E(h\cdot Q) & \xrightarrow{E(h\restriction P)+E(h\restriction Q)} & E(P)+E(Q) \\
{\scriptstyle E(s\restriction h\cdot R)+E(s\restriction h\cdot S)}\downarrow & & \downarrow{\scriptstyle E(s\restriction R)+E(s\restriction S)} \\
E(h\cdot R)+E(h\cdot S) & \xrightarrow[+E(h\restriction S)]{E(h\restriction R)} & E(R)+E(S) \\
{\scriptstyle E(t\restriction h\cdot R)+E(t\restriction h\cdot S)}\uparrow & & \uparrow{\scriptstyle E(t\restriction R)+E(t\restriction S)} \\
E(h\cdot P')+E(h\cdot Q') & \xrightarrow[+E(h\restriction Q')]{E(h\restriction P')} & E(P')+E(Q'),
\end{array}$$

which commutes by induction hypothesis.
Case $in_P^{a,\nu b}$**.**

$$\begin{array}{ccc}
E(h\cdot a(b).P) & \xrightarrow{E(h\restriction a(b).P)} & E(a(b).P) \\
{\scriptstyle E(s\restriction(h,(b\mapsto b'))\cdot in_P^{a,\nu b})}\downarrow & & \downarrow{\scriptstyle E(s\restriction in_P^{a,\nu b})} \\
E((h,(b\mapsto b'))\cdot in_P^{a,\nu b}) & \xrightarrow[E((h,(b\mapsto b'))\restriction in_P^{a,\nu b})]{} & E(in_P^{a,\nu b}) \\
{\scriptstyle E(t\restriction(h,(b\mapsto b'))\cdot in_P^{a,\nu b})}\uparrow & & \uparrow{\scriptstyle E(t\restriction in_P^{a,\nu b})} \\
E((h+(b\mapsto b'))\cdot P) & \xrightarrow[E((h+(b\mapsto b'))\restriction P)]{} & E(P)
\end{array}$$

unfolds to

$$\begin{array}{ccc}
E((h+(b\mapsto a_{\gamma'}))\cdot P) & \xrightarrow{E(h+(a_\gamma\mapsto a_{\gamma'})\restriction P)} & E((b\mapsto a_\gamma)\cdot P) \\
{\scriptstyle E((b'\mapsto a_{\gamma'})\restriction P')}\downarrow & & \downarrow{\scriptstyle E((b\mapsto a_\gamma)\restriction P)} \\
E((h+(b\mapsto b'))\cdot P) & \xrightarrow{E(h+(b\mapsto b')\restriction P)} & E(P) \\
\| & & \| \\
E((h+(b\mapsto b'))\cdot P) & \xrightarrow[E(h+(b\mapsto b')\restriction P)]{} & E(P),
\end{array}$$

where $P' = (h+(b\mapsto b'))\cdot P$, which commutes by Lemma B.1.
Case $\nabla b.R$**.**

$$\begin{array}{ccc}
E(h\cdot \nu b.P) & \xrightarrow{E(h\restriction \nu b.P)} & E(\nu b.P) \\
{\scriptstyle E(s\restriction(h,(b\mapsto b'))\cdot\nabla b.R)}\downarrow & & \downarrow{\scriptstyle E(s\restriction\nabla b.R)} \\
E((h,(b\mapsto b'))\cdot\nabla b.R) & \xrightarrow[E((h,(b\mapsto b'))\restriction\nabla b.R)]{} & E(\nabla b.R) \\
{\scriptstyle E(t\restriction(h,(b\mapsto b'))\cdot\nabla b.R)}\uparrow & & \uparrow{\scriptstyle E(t\restriction\nabla b.R)} \\
E((h+(b\mapsto b'))\cdot Q) & \xrightarrow[E((h+(b\mapsto b'))\restriction Q)]{} & E(Q)
\end{array}$$

unfolds to

$$\begin{array}{ccc}
E((h+(b\mapsto a_{\gamma'}))\cdot P) & \xrightarrow{E(h+(a_\gamma\mapsto a_{\gamma'})\restriction P')} & E((b\mapsto a_\gamma)\cdot P) \\
{\scriptstyle E((b'\mapsto a_{\gamma'})\restriction P'')}\downarrow & & \downarrow{\scriptstyle E((b\mapsto a_\gamma)\restriction P)} \\
E((h+(b\mapsto b'))\cdot P) & \xrightarrow{E(h+(b\mapsto b')\restriction P)} & E(P) \\
{\scriptstyle E(s\restriction(h+(b\mapsto b'),id)\cdot R)}\downarrow & & \downarrow{\scriptstyle E(s\restriction R)} \\
E((h+(b\mapsto b'),id)\cdot R) & \xrightarrow[E((h+(b\mapsto b'),id)\restriction R)]{} & E(R) \\
{\scriptstyle E(t\restriction(h+(b\mapsto b'),id)\cdot R)}\uparrow & & \uparrow{\scriptstyle E(t\restriction R)} \\
E((h+(b\mapsto b'))\cdot Q) & \xrightarrow[h+(b\mapsto b')\restriction Q]{} & E(Q),
\end{array}$$

where $P' = (b \mapsto a_\gamma) \cdot P$ and $P'' = (h + (b \mapsto b')) \cdot P$, which commutes by Lemma B.1 (top square) and induction hypothesis.

Case $R \rhd^{va_\gamma} S$ ($S \lhd^{va_\gamma} R$ is symmetric).

$$\begin{array}{ccc}
E(h \cdot (P|Q)) & \xrightarrow{E(h \restriction (P|Q))} & E(P|Q) \\
{\scriptstyle E(s \restriction (h,id) \cdot (R \rhd^{va_\gamma} S))}\downarrow & & \downarrow{\scriptstyle E(s \restriction R \rhd^{va_\gamma} S)} \\
E((h,id) \cdot (R \rhd^{va_\gamma} S)) & \xrightarrow{E((h,id) \restriction R \rhd^{va_\gamma} S)} & E(R \rhd^{va_\gamma} S) \\
{\scriptstyle E(t \restriction (h,id) \cdot R \rhd^{va_\gamma} S)}\uparrow & & \uparrow{\scriptstyle E(t \restriction R \rhd^{va_\gamma} S)} \\
E(h \cdot va_{\gamma,\delta}.(P'|Q')) & \xrightarrow[E(h \restriction va_{\gamma,\delta}.(P'|Q'))]{} & E(va_{\gamma,\delta}.(P'|Q'))
\end{array}$$

unfolds to

$$\begin{array}{ccc}
E(h \cdot P) + E(h \cdot Q) & \xrightarrow{E(h \restriction P) + E(h \restriction Q)} & E(P) + E(Q) \\
\downarrow & & \downarrow{\scriptstyle E(s \restriction R) + E(s \restriction S)} \\
\begin{array}{r} E(s \restriction (h, (a_\gamma \mapsto a_{\gamma'})) \cdot R) \\ + E(s \restriction (h, (a_\gamma \mapsto a_{\gamma'})) \cdot S) \end{array} & & \\
\downarrow & & \\
\begin{array}{r} E((h, (a_\gamma \mapsto a_{\gamma'})) \cdot R) \\ + E((h, (a_\gamma \mapsto a_{\gamma'})) \cdot S) \end{array} & \xrightarrow[+ E((h,(a_\gamma \mapsto a_{\gamma'})) \restriction S)]{E((h,(a_\gamma \mapsto a_{\gamma'})) \restriction R)} & E(R) + E(S) \\
\uparrow & & \uparrow{\scriptstyle E(t \restriction R) + E(t \restriction S)} \\
\begin{array}{r} E(t \restriction (h, (a_\gamma \mapsto a_{\gamma'})) \cdot R) \\ + E(t \restriction (h, (a_\gamma \mapsto a_{\gamma'})) \cdot S) \end{array} & & \\
\uparrow & & \\
\begin{array}{r} E((h + (a_\gamma \mapsto a_{\gamma'})) \cdot P') \\ + E((h + (a_\gamma \mapsto a_{\gamma'})) \cdot Q') \end{array} & \xrightarrow[+ E(h + (a_\gamma \mapsto a_{\gamma'}) \restriction Q')]{E(h + (a_\gamma \mapsto a_{\gamma'}) \restriction P')} & E(P') + E(Q'),
\end{array}$$

where $P' = (h + (b \mapsto b')) \cdot P$, which commutes by induction hypothesis.

Case $va_{\gamma,\delta}.R$.

$$\begin{array}{ccc}
E(h \cdot va_{\gamma,\delta}.P) & \xrightarrow{E(h \restriction va_{\gamma,\delta}.P)} & E(va_{\gamma,\delta}.P) \\
{\scriptstyle E(s \restriction (h,k) \cdot va_{\gamma,\delta}.R)}\downarrow & & \downarrow{\scriptstyle E(s \restriction va_{\gamma,\delta}.R)} \\
E((h,k) \cdot va_{\gamma,\delta}.R) & \xrightarrow{E((h,k) \restriction va_{\gamma,\delta}.R)} & E(va_{\gamma,\delta}.R) \\
{\scriptstyle E(t \restriction (h,k) \cdot va_{\gamma,\delta}.R)}\uparrow & & \uparrow{\scriptstyle E(t \restriction va_{\gamma,\delta}.R)} \\
E(h \cdot va_{\gamma,\delta}.Q) & \xrightarrow[E(h \restriction va_{\gamma,\delta}.Q)]{} & E(va_{\gamma,\delta}.Q)
\end{array}$$

unfolds to

$$\begin{array}{ccc}
E((h+(a_{\gamma,\delta}\mapsto a_{\gamma'}))\cdot P) & \xrightarrow{E(h+(a_{\gamma'}\mapsto a_{\gamma'})\restriction P')} & E((a_{\gamma,\delta}\mapsto a_\gamma)\cdot P) \\
{\scriptstyle E((a_{\gamma',\delta'}\mapsto a_{\gamma'})\restriction P'')}\downarrow & & \downarrow{\scriptstyle E((a_{\gamma,\delta}\mapsto a_\gamma)\restriction P)} \\
E((h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}))\cdot P) & \xrightarrow{E(h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'})\restriction P)} & E(P) \\
{\scriptstyle E(s\restriction(h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}),k)\cdot R)}\downarrow & & \downarrow{\scriptstyle E(s\restriction R)} \\
E((h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}),k)\cdot R) & \xrightarrow{E((h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}),k)\restriction R)} & E(R) \\
{\scriptstyle E(t\restriction(h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}),k)\cdot R)}\uparrow & & \uparrow{\scriptstyle E(t\restriction R)} \\
E((h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}))\cdot Q) & \xrightarrow{E(h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'})+k\restriction Q)} & E(Q),
\end{array}$$

where $P' = (a_{\gamma,\delta} \mapsto a_\gamma)\cdot P$ and $P'' = (h+(a_{\gamma,\delta}\mapsto a_{\gamma',\delta'}))\cdot P$, which commutes by Lemma B.1 (top square) and by induction hypothesis.

Case $L|Q$ ($P|R$ is symmetric).

$$\begin{array}{ccc}
E(h\cdot(P|Q)) & \xrightarrow{E(h\restriction(P|Q))} & E(P|Q) \\
{\scriptstyle E(s\restriction(h,k)\cdot(L|Q))}\downarrow & & \downarrow{\scriptstyle E(s\restriction L|Q)} \\
E((h,k)\cdot(L|Q)) & \xrightarrow{E((h,k)\restriction L|Q)} & E(L|Q) \\
{\scriptstyle E(t\restriction(h,k)\cdot L|Q)}\uparrow & & \uparrow{\scriptstyle E(t\restriction L|Q)} \\
E((h+k)\cdot(P'|Q)) & \xrightarrow{E((h+k)\restriction P'|Q)} & E(P'|Q)
\end{array}$$

unfolds to

$$\begin{array}{ccc}
E(h\cdot P)+E(h\cdot Q) & \xrightarrow{E(h\restriction P)+E(h\restriction Q)} & E(P)+E(Q) \\
{\scriptstyle E(s\restriction(h,k)\cdot L)+id}\downarrow & & \downarrow{\scriptstyle E(s\restriction R)+id} \\
E((h,k)\cdot L)+E(h\cdot Q) & \xrightarrow{E((h,k)\restriction L)+E(h\restriction Q)} & E(L)+E(Q) \\
{\scriptstyle E(t\restriction(h,k)\cdot L)+E(i^{\delta'}_{\gamma'}\restriction h\cdot Q)}\uparrow & & \uparrow{\scriptstyle E(t\restriction L)+E(i^{\delta}_{\gamma}\restriction Q)} \\
E((h+k)\cdot P')+E(i^{\delta'}_{\gamma'}\cdot h\cdot Q) & \xrightarrow[+E(h+k\restriction i^{\delta}_{\gamma}\cdot Q)]{E(h+k\restriction P')} & E(P')+E(i^{\delta}_{\gamma}\cdot Q),
\end{array}$$

which makes sense because $(h+k)\circ i^{\delta}_{\gamma} = i^{\delta'}_{\gamma'}\circ h$, and commutes by induction hypothesis and Lemma B.1 (bottom square, right-hand term of sums). □

Altogether, we have proved:

Proposition B.3. *E is indeed a functor $el(\mathscr{T}^{+}_{\pi}(1)) \to \widehat{\mathbb{F}}$.*