Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science 207 (2008) 89–106
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

An Exercise on Transition Systems

Paula R. Ribeiro[1] L. S. Barbosa[2]

Department of Informatics
Minho University, Portugal

Shuling Wang[3]

LMAM and Department of Informatics, School of Mathematical Sciences
Peking University, Beijing, China

Abstract

Labelled transition systems admit different but equivalent characterizations either as relational structures or coalgebras for the powerset functor, each of them with their own merits. Notions of simulation and bisimulation, for example, are expressed in the pointfree relational calculus in a very concise and precise way. On the other hand, the coalgebraic perspective regards processes as inhabitants of a final universe and allows for an intuitive definition of the semantics of process' combinators.
This paper is an exercise on such a dual characterisation. In particular, it discusses how a notion of weak bisimilarity can be lifted from the relational to the coalgebraic level, to become an effective reasoning tool on coinductively defined process algebras.

Keywords: process algebra, bisimularity, coalgebraic, relational.

1 Introduction

Recent approaches to process calculi semantics build on representations of *labelled transition systems* as coalgebras for (some combinations of) the powerset functor. Such coalgebraic characterizations not only provide a generic setting for fundamental constructions (*e.g.*, bisimulation regarded as equality in the final coalgebra), but also makes it easier to generalize typical transition systems concepts to broader classes of dynamic systems (*e.g.*, probabilistic automata [12,2] or hybrid systems [11]).

In this context, references [6,7] introduced a denotational approach to the *design* of process algebras in which processes are identified with inhabitants of a final

[1] Email: priobom@gmail.pt
[2] Email: lsb@di.uminho.pt
[3] Email: joycy@math.pku.edu.cn

coalgebra and their combinators defined by coinductive extension (of 'one-step' behaviour generator functions). The *universality* of such constructions entails both definitional and proof principles on top of which the development of the whole calculus is based. Combined with the *pointfree* 'calculational' style entailed by category theory, this leads to a generic way of reasoning about processes in which, in particular, proofs by bisimulation, which classicaly involve the explicit construction of such a relation [15], are replaced by *equational reasoning* [4].

In this approach, transition systems over a set A of labels, classicaly specified as *binary relations*

$$\underset{\alpha}{\longleftarrow} : A \times U \longleftarrow U \tag{1}$$

are given by coalgebras

$$\alpha : \mathcal{P}(A \times U) \longleftarrow U \tag{2}$$

for $\mathcal{P}(A \times \mathsf{Id})$, where $\mathcal{P}$ and Id denote, respectively, the (finite) powerset and the identity functor. It is well-known that set-valued functions, such as coalgebra (2) are models of binary relations and, conversely, any such relation is uniquely transposed into a set-valued function. The existence and uniqueness of such a transformation leads to the identification of a *transpose* operator Λ [8] characterized by an universal property which, for this particular case, becomes

$$\alpha = \Lambda \underset{\alpha}{\longleftarrow} \;\equiv\; \underset{\alpha}{\longleftarrow} = \in \cdot\, \alpha \tag{3}$$

Moreover, whenever $\mathcal{P}$ in (2) is restricted to the *finite* powerset, to enforce the existence of a final universe, equivalence (3) establishes again a bijective correspondence between the resulting coalgebras and *image finite* relations. In any case, the fundamental observation is that, the transpose being an isomorphism, one may reason either in one side of equivalence (3) or in the other, whichever offers a richer setting for calculation.

This paper reports on such an exercise: seeking for a suitable definition of *weak bisimulation* within the coalgebraic setting mentioned above, we realized that such equivalences would be described in more intuitive way in a relational setting. Constructions are then translated back to the coalgebraic level, where they add up to the coinductive, equational calculational toolkit. In a sense, as discussed elsewhere [17], the role of the powerset transpose is similar to that of the Laplace transform to reduce arbitrary expressions to a polynomial format. We believe this way of proceeding is not unrelated to this workshop aim of harnessing theories for supporting software construction.

A subsidiary objective of this paper is to show that, irrespective of the (coalgebraic or relational) level of expression, effective reasoning requires a 'concise and precise' notation and an expressive calculus. In most cases this entails going *pointfree*, thus replacing application by composition. For example, in the relational setting, keeping track of nested quantified variables quickly becomes a nightmare.

[4] In the *dual* world of functional programming the role of such 'universals' is the basis of a whole discipline of algorithm derivation and transformation, which can be traced back to the so-called *Bird-Meertens formalism* [9] and the foundational work of T. Hagino [10].

Therefore the paper is formulated within the pointfree calculus of binary relations as developed in, *e.g.*, [4] and the second part of [8].

The paper is organized as follows. Section 2, provides the basic backgound for the paper. In particular, it recalls the rudiments of the relational calculus, introduces notation and briefly reviews the calculational approach to process calculi introduced elsewhere by the first author. Section 3 revisits the relationship connecting the coalgebraic and the relational levels and provides a pointfree relational account of simulation and bisimulation. Characterizing the weak versions of such concepts is the purpose of section 4 which contains the main contribution of the paper. The proposed approach is illustrated by a number of examples. In particular, we show how process properties formulated in terms of weak bisimulation can still be proved in the equational, pointfree style used in [6] and popularised in the 'mathematics of program construction community'. Finally, section 5 summarizes what has been achieved and enumerates a few research questions for the future.

2 Background

Although the paper resorts to a quite standard mathematical notation to express sets, functions and relations, this section fixes some notation and reviews a few basic notions. Furthermore, it contains a brief introduction to the coinductive approach to process algebra design proposed in [6,7], which provides the context for this paper.

2.1 Relations

Let $R : B \longleftarrow A$ denote a binary relation on (source) type A and (target) type B, and bRa stand for the representation of $\langle b, a\rangle \in R$. The set of relations from A to B is *ordered* by inclusion $\subseteq$, with relation equality being established by anti-symmetry. Fact $R \subseteq S$ means that relation S is either more defined or less deterministic than R, that is, for all a and b of the appropriate types, $bRa \Rightarrow bSa$.

The algebra of relations is built on top of three basic operators: composition $(R \cdot S)$, meet $(R \cap S)$ and converse (R°). As expected, $aR^\circ b$ iff bRa, meet corresponds to set-theorectical intersection and $\cdot$ generalizes functional composition: $b(R \cdot S)c$ holds iff there exists some $a \in A$ such that $bRa \wedge aSc$.

Any function f can be seen as the relation given by its graph, which, in this paper, is also denoted by f. Therefore $bfa \;\equiv\; b = fa$. In this setting functions enjoy a number of properties of which the following is singled out by its role in the pointwise to pointfree conversion:

$$b\,(f^\circ \cdot R \cdot g)\,a \;\equiv\; (fb)\,R\,(ga) \tag{4}$$

Conversely, any relation $R : B \longleftarrow A$ can be uniquely transposed into a set-valued function $\Lambda R : \mathcal{P}B \longleftarrow A$, where the transpose operator Λ satisfies the following universal property: $f = \Lambda R \;\equiv\; (bRa \;\equiv\; b \in (fa))$. The interplay between functions and relations is also captured by the so-called *shunting* laws [4], of which the following will be used in the paper:

$$f \cdot R \subseteq S \cdot g \;\equiv\; R \subseteq f^{\circ} \cdot S \cdot g \tag{5}$$

Several constructions in the relational calculus emerge as *Galois connections* [3], $C \dashv C'$. Such is the case, for example, of the left and right *division* operators given, respectively by

$$\cdot R \dashv /R \quad \text{and} \quad R\cdot \dashv R\backslash$$

References [8] and, mainly, [4], provide a detailed account of the calculus of binary relations, in a *pointfree* calculational style.

2.2 Processes

Technically, the coinductive reconstruction of classical process algebra proposed in [6,7] amounts to the systematic use of the universal property of coinductive extension, i.e., the existence, for each arbitrary T-coalgebra $\langle U, p : \mathsf{T}\,U \longleftarrow U\rangle$, of a unique morphism $[\![(p)]\!]$ to the final coalgebra $\omega_{\mathsf{T}} : \mathsf{T}\,\nu_{\mathsf{T}} \longleftarrow \nu_{\mathsf{T}}$ satisfying

$$k = [\![(p)]\!]_{\mathsf{T}} \quad\Leftrightarrow\quad \omega_{\mathsf{T}} \cdot k = \mathsf{T}\,k \cdot p \tag{6}$$

Such $[\![(p)]\!]$, which, in the tradition of [14] or [8] is referred to as the p *anamorphism*, represents the behaviour generated by p and comes equipped with a bunch of laws usefull in calculation [5].

As explained in the previous section, in [6] processes are regarded as inhabitants of the final coalgebra $\omega : \mathcal{P}(Act \times \nu) \longleftarrow \nu$, whose carrier is the set of possibly infinite labelled trees, finitely branched and quotiented by the greatest bisimulation [1]. On top of it process combinators are defined. Typically, the so-called *dynamic* combinators, *i.e.*, combinators which are 'consumed' on action occurrence, have a direct definition in terms of the available observations. For example, *inaction* is represented as a constant $\mathsf{nil} : \nu \longleftarrow \mathbf{1}$ upon which no relevant observation can be made, *i.e.*, $\omega \cdot \mathsf{nil} = \underline{\emptyset}$. Prefix gives rise to an Act-indexed family of operators $a. : \nu \longleftarrow \nu$, with $a \in A$, whereas the possible actions of the non deterministic choice of two processes p and q corresponds to the collection of all actions allowed for p and q. Formally, $\omega \cdot + \;=\; \cup \cdot (\omega \times \omega)$ and $\omega \cdot a. \;=\; \mathsf{sing} \cdot \mathsf{label}_a$, where $\mathsf{sing} = \lambda x\,.\;\{x\}$ and $\mathsf{label}_a = \lambda x\,.\;\langle a, x\rangle$.

On the other hand, *static* combinators, which persist over action occurrence, being recursive, are defined as anamorphisms. An example, used later in the paper, is *interleaving* $||| : \nu \longleftarrow \nu \times \nu$ which represents an interaction-free form of parallel composition. The following definition captures the intuition that the observations over the interleaving of two processes correspond to all possible interleavings of

[5] The *existence* assertion underlying (6) (corresponding to the left to right implication) provides a *definition* principle for (circular) functions to the final coalgebra which amounts to equip their source with a coalgebraic structure — the *gene* — specifying the *next-step* dynamics. The *uniqueness* part, underlying right to left implication in (6), on the other hand, entails *coinduction* as a proof principle.

observations of their arguments. Thus, $||| \;=\; [\!(\alpha_{|||})\!]$, where [6]

$$
\begin{aligned}
\alpha_{|||} \;=\; \nu \times \nu &\xrightarrow{\vartriangle} (\nu \times \nu) \times (\nu \times \nu) \xrightarrow{(\omega \times \mathsf{id}) \times (\mathsf{id} \times \omega)} (\mathcal{P}(Act \times \nu) \times \nu) \times (\nu \times \mathcal{P}(Act \times \nu)) \\
&\xrightarrow{\tau_r \times \tau_l} \mathcal{P}(Act \times (\nu \times \nu)) \times \mathcal{P}(Act \times (\nu \times \nu)) \xrightarrow{\cup} \mathcal{P}(Act \times (\nu \times \nu))
\end{aligned}
$$

The interested reader is refered to [6,7] for the full development of process calculi along the lines just sketeched.

3 Bisimulation Revisited

3.1 The Relational Transpose

The relational transpose (3) is the basic tool to swap from the coalgebraic into the relational setting, or vice-versa. In section 4 this will be applied to discuss weak bisimulation for coinductively defined processes For the moment, however, we shall revisit the notion of bisimulation speaking the language of relations. On the one hand this is necessary to pave the way to section 4; on the other it provides an interesting example of how equational pointfree reasoning style actually simplifies relational proofs.

Our first step is to show that *transposition* extends to morphisms. Recall that a morphism $h : \beta \longleftarrow \alpha$ from coalgebra α to β is a function between the corresponding state spaces preserving the dynamics of the source coalgebra, *i.e.*, such that

$$\mathcal{P}(\mathsf{id} \times h) \cdot \alpha \;=\; \beta \cdot h \tag{7}$$

What is the relational counterpart of this equation? The answer is given by an easy calculation.

Lemma 3.1 *A function $h : V \longleftarrow U$ is a morphism relating two $\mathcal{P}(A \times \mathsf{Id})$-coalgebras, α and β, defined over U and V, respectively, if and only if*

$$(\mathsf{id} \times h) \cdot {}_{\alpha}\!\longleftarrow \;=\; {}_{\beta}\!\longleftarrow \cdot\, h \tag{8}$$

whose formulation involves the corresponding transition systems ${}_{\alpha}\!\longleftarrow$ and ${}_{\beta}\!\longleftarrow$.

Proof.

[6] Morphisms $\tau_r : \mathcal{P}(Act \times (X \times C)) \longleftarrow \mathcal{P}(Act \times X) \times C$ and $\tau_l : \mathcal{P}(Act \times (C \times X)) \longleftarrow C \times \mathcal{P}(Act \times X)$ stand for, respectively, the right and left *strength* associated to functor $\mathcal{P}(Act \times \mathsf{Id})$.

$$(\mathsf{id} \times h) \cdot {}_{\alpha}\!\longleftarrow \;=\; {}_{\beta}\!\longleftarrow \cdot h$$
$$\equiv \quad \{ \ \Lambda \text{ is an isomorphism } \}$$
$$\Lambda((\mathsf{id} \times h) \cdot {}_{\alpha}\!\longleftarrow) \;=\; \Lambda({}_{\beta}\!\longleftarrow \cdot h)$$
$$\equiv \quad \{ \ \Lambda(f \cdot R) = \mathcal{P}f \cdot \Lambda R \ \text{ and } \ \Lambda(R \cdot f) = \Lambda R \cdot f \ \}$$
$$\mathcal{P}(\mathsf{id} \times h) \cdot \Lambda({}_{\alpha}\!\longleftarrow) \;=\; \Lambda({}_{\beta}\!\longleftarrow) \cdot h$$
$$\equiv \quad \{ \ \text{definition of } {}_{\alpha}\!\longleftarrow \ \}$$
$$\mathcal{P}(\mathsf{id} \times h) \cdot \Lambda(\in \cdot \alpha) \;=\; \Lambda(\in \cdot \beta) \cdot h$$
$$\equiv \quad \{ \ \Lambda(R \cdot f) = \Lambda R \cdot f \ \}$$
$$\mathcal{P}(\mathsf{id} \times h) \cdot \Lambda(\in) \cdot \alpha \;=\; \Lambda(\in) \cdot \beta \cdot h$$
$$\equiv \quad \{ \ \Lambda(\in) = \mathsf{id} \ \}$$
$$\mathcal{P}(\mathsf{id} \times h) \cdot \alpha \;=\; \beta \cdot h$$

□

Representing ${}_{\alpha}\!\longleftarrow$ as an A-indexed family of binary relations ${}_{\alpha}\!\xleftarrow{a} : U \longleftarrow U$ for all $a \in A$ [7], equation (8) reads

$$h \cdot {}_{\alpha}\!\xleftarrow{a} \;=\; {}_{\beta}\!\xleftarrow{a} \cdot h \tag{9}$$

For each $a \in A$, (9) can be decomposed into the conjuntion of two inclusions:

$$h \cdot {}_{\alpha}\!\xleftarrow{a} \;\subseteq\; {}_{\beta}\!\xleftarrow{a} \cdot h \tag{10}$$
$${}_{\beta}\!\xleftarrow{a} \cdot h \;\subseteq\; h \cdot {}_{\alpha}\!\xleftarrow{a} \tag{11}$$

which, once turned into predicates and made pointwise, adopt the more familiar form

$$\forall_{u,u' \in U} \, . \ u' \ {}_{\alpha}\!\xleftarrow{a} \ u \ \Rightarrow \ h\,u' \ {}_{\beta}\!\xleftarrow{a} \ h\,u \tag{12}$$
$$\forall_{u \in U, v' \in V} \, . \ v' \ {}_{\beta}\!\xleftarrow{a} \ h\,u \ \Rightarrow \ \exists_{u' \in U} . \ u' \ {}_{\alpha}\!\xleftarrow{a} \ u \ \wedge \ v' = h\,u' \tag{13}$$

Proof.

[7] For notational convinenence the converse of this relation will be written as $({}_{\alpha}\!\xleftarrow{a})^{\circ} = \xrightarrow{a}_{\alpha}$.

$$h \cdot {}_{\alpha}\overset{a}{\longleftarrow} \;\subseteq\; {}_{\beta}\overset{a}{\longleftarrow} \cdot h$$

$\equiv$ { *shunting* rule (5) }

$${}_{\alpha}\overset{a}{\longleftarrow} \;\subseteq\; h^{\circ} \cdot {}_{\beta}\overset{a}{\longleftarrow} \cdot h$$

$\equiv$ { going pointwise }

$$\forall_{u,u' \in U} \,.\, u' \;{}_{\alpha}\overset{a}{\longleftarrow}\; u \;\Rightarrow\; u' \,(h^{\circ} \cdot {}_{\beta}\overset{a}{\longleftarrow} \cdot h)\, u$$

$\equiv$ { law (4) }

$$\forall_{u,u' \in U} \,.\, u' \;{}_{\alpha}\overset{a}{\longleftarrow}\; u \;\Rightarrow\; h\,u' \;{}_{\beta}\overset{a}{\longleftarrow}\; h\,u$$

and

$${}_{\beta}\overset{a}{\longleftarrow} \cdot h \;\subseteq\; h \cdot {}_{\alpha}\overset{a}{\longleftarrow}$$

$\equiv$ { going pointwise }

$$\forall_{u \in U, v' \in V} \,.\, v' \,(\,{}_{\beta}\overset{a}{\longleftarrow} \cdot h)\, u \;\Rightarrow\; v' \,(h \cdot {}_{\alpha}\overset{a}{\longleftarrow})\, u$$

$\equiv$ { law (4) and relational composition }

$$\forall_{u \in U, v' \in V} \,.\, v' \;{}_{\beta}\overset{a}{\longleftarrow}\; h\,u \;\Rightarrow\; \exists_{u' \in U} .\, u' \;{}_{\alpha}\overset{a}{\longleftarrow}\; u \;\wedge\; v' = h\,u')$$

□

Taken jointly these equations express that, not only the dynamics of α (represented by transition system ${}_{\alpha}\longleftarrow$) is *preserved* by h (10), but also the dynamics of β is reflected backwards through the same h (11). This leads us directly to bisimulations.

3.2 Simulation and Bisimulation

The classical definition of simulation, as given *e.g.* in [15], reads as follows:

Definition 3.2 Given transition systems ${}_{\alpha}\longleftarrow : U \times A \longleftarrow U$ and ${}_{\beta}\longleftarrow : V \times A \longleftarrow V$ over the same label set A, a *simulation* of ${}_{\alpha}\longleftarrow$ in ${}_{\beta}\longleftarrow$ is a relation $S : V \longleftarrow U$ such that

$$\forall_{a \in A} \forall_{u \in U, v \in V} \,.\, vSu \;\Rightarrow\; (\forall_{u' \in U} \,.\, u' \;{}_{\alpha}\overset{a}{\longleftarrow}\; u \Rightarrow (\exists_{v' \in V} \,.\, v' \;{}_{\beta}\overset{a}{\longleftarrow}\; v \;\wedge\; v'Su')) \tag{14}$$

This definition can be rephrased in a form in which the first order expression is turned into a purely algebraic expression:

Lemma 3.3 *A relation $S : V \longleftarrow U$ is a* simulation *of* ${}_{\alpha}\longleftarrow$ *in* ${}_{\beta}\longleftarrow$ *iff, for all* $a \in A$

$$S \cdot \xrightarrow{a}_{\alpha} \subseteq \xrightarrow{a}_{\beta} \cdot S \tag{15}$$

Proof.

$$\forall_{a\in A}\forall_{u\in U,v\in V} .\ vSu \ \Rightarrow\ (\forall_{u'\in U} .\ u'\ {}_{\alpha}\overset{a}{\longleftarrow}\ u \Rightarrow (\exists_{v'\in V} .\ v'\ {}_{\beta}\overset{a}{\longleftarrow}\ v \wedge v'Su'))$$

$\equiv$ { definition of relational composition }

$$\forall_{a\in A}\forall_{u\in U,v\in V} .\ vSu \ \Rightarrow\ (\forall_{u'\in U} .\ u \overset{a}{\longrightarrow}_{\alpha} u' \Rightarrow v(\overset{a}{\longrightarrow}_{\beta} \cdot S)u'$$

$\equiv$ { definition of left relational division }

$$\forall_{a\in A}\forall_{u\in U,v\in V} .\ vSu \ \Rightarrow\ v((\overset{a}{\longrightarrow}_{\beta} \cdot S)/ \overset{a}{\longrightarrow}_{\alpha})u$$

$\equiv$ { going pointfree }

$$S \subseteq (\overset{a}{\longrightarrow}_{\beta} \cdot S)/ \overset{a}{\longrightarrow}_{\alpha}$$

$\equiv$ { Galois connection: $(\cdot R) \dashv (/R)$ }

$$S \cdot \overset{a}{\longrightarrow}_{\alpha} \subseteq \overset{a}{\longrightarrow}_{\beta} \cdot S$$

□

Using equation (15) the proof of the following folklore result becomes rather concise: no more than 3 steps in each derivation.

Lemma 3.4 *(1) The empty relation ($\perp$) and identity (*id*) are simulations. (2) The composition and (3) the union of two simulations is still a simulation.*

Proof.

(i) Let ${}_{\alpha}\longleftarrow$ be a transition system over A and state space U. Then

$$\perp \cdot \overset{a}{\longrightarrow}_{\alpha} \subseteq \overset{a}{\longrightarrow}_{\beta} \cdot \perp \ \wedge\ \mathsf{id} \cdot \overset{a}{\longrightarrow}_{\alpha} \subseteq \overset{a}{\longrightarrow}_{\alpha} \cdot \mathsf{id}$$

$\equiv$ { $\perp$ and id are, respectively the zero and identity element of $\cdot$ }

true

(ii) Consider, now, ${}_{\beta}\longleftarrow : V\times A \longleftarrow V$, ${}_{\gamma}\longleftarrow : Z\times A \longleftarrow Z$ and ${}_{\alpha}\longleftarrow : U\times A \longleftarrow U$, where simulations $S : {}_{\beta}\longleftarrow \longleftarrow {}_{\gamma}\longleftarrow$ and $R : {}_{\gamma}\longleftarrow \longleftarrow {}_{\alpha}\longleftarrow$ are defined. Then,

$$(S \cdot R) \cdot \overset{a}{\longrightarrow}_{\alpha} \subseteq \overset{a}{\longrightarrow}_{\beta} \cdot (S \cdot R)$$

$\Leftarrow$ { $S \cdot \overset{a}{\longrightarrow}_{\gamma} \subseteq \overset{a}{\longrightarrow}_{\beta} \cdot S$, $R \cdot \overset{a}{\longrightarrow}_{\alpha} \subseteq \overset{a}{\longrightarrow}_{\gamma} \cdot R$, $\cdot$-assoc, monotony }

$$(S \cdot R) \cdot \overset{a}{\longrightarrow}_{\alpha} \subseteq (S \cdot R) \cdot \overset{a}{\longrightarrow}_{\alpha}$$

$\equiv$ { *trivial* }

true

(iii) Consider, now, systems ${}_{\beta}\longleftarrow : V \times A \longleftarrow V$ and ${}_{\alpha}\longleftarrow : U \times A \longleftarrow U$ connected by simulations $S : {}_{\beta}\longleftarrow \longleftarrow {}_{\alpha}\longleftarrow$ and $R : {}_{\beta}\longleftarrow \longleftarrow {}_{\alpha}\longleftarrow$. Then,

$$
\begin{array}{ll}
 & (S \cup R) \cdot \xrightarrow{a}_{\alpha} \subseteq \xrightarrow{a}_{\beta} \cdot (S \cup R) \\
\equiv & \{ \ (R\cdot) \text{ and } (\cdot R) \text{ as lower adjoints preserve } \cup \ \} \\
 & (S \cdot \xrightarrow{a}_{\alpha} \cup R \cdot \xrightarrow{a}_{\alpha}) \subseteq (\xrightarrow{a}_{\beta} \cdot S \cup \xrightarrow{a}_{\beta} \cdot R) \\
\Leftarrow & \{ \ \cup \text{ definition } \} \\
 & S \cdot \xrightarrow{a}_{\alpha} \subseteq \xrightarrow{a}_{\beta} \cdot S \ \wedge \ R \cdot \xrightarrow{a}_{\alpha} \subseteq \xrightarrow{a}_{\beta} \cdot R \\
\equiv & \{ \ \text{hypothesis} \ \} \\
 & \mathsf{true}
\end{array}
$$

□

The standard definition of bisimulation — a relation S such that S itself and its converse S° are both simulations — can also be rephrased as follows:

Lemma 3.5 *A relation $S : V \longleftarrow U$ is a* bisimulation *between ${}_{\alpha}\!\longleftarrow$ and ${}_{\beta}\!\longleftarrow$ iff*

$$S \cdot \xrightarrow{a}_{\alpha} \subseteq \xrightarrow{a}_{\beta} \cdot S \quad \wedge \quad {}_{\beta}\!\xleftarrow{a} \cdot S \subseteq S \cdot {}_{\alpha}\!\xleftarrow{a} \tag{16}$$

for all $a \in A$.

Proof. The first conjunct is the definition of S as a simulation. For the second

$$
\begin{array}{ll}
 & S^\circ \ \text{ is a simulation} \\
\equiv & \{ \ \text{defining equation (15)} \ \} \\
 & S^\circ \cdot \xrightarrow{a}_{\beta} \subseteq \xrightarrow{a}_{\alpha} \cdot S^\circ \\
\equiv & \{ \ (\xrightarrow{a}_{\gamma})^\circ = {}_{\gamma}\!\xleftarrow{a} \ \} \\
 & S^\circ \cdot ({}_{\beta}\!\xleftarrow{a})^\circ \subseteq ({}_{\alpha}\!\xleftarrow{a})^\circ \cdot S^\circ \\
\equiv & \{ \ (R \cdot S)^\circ = S^\circ \cdot R^\circ \ \} \\
 & ({}_{\beta}\!\xleftarrow{a} \cdot S)^\circ \subseteq (S \cdot {}_{\alpha}\!\xleftarrow{a})^\circ \\
\equiv & \{ \ \text{converse is monotonic: } R \subseteq S \ \equiv \ R^\circ \subseteq S^\circ \ \} \\
 & {}_{\beta}\!\xleftarrow{a} \cdot S \subseteq S \cdot {}_{\alpha}\!\xleftarrow{a}
\end{array}
$$

□

Introducing variables, we quickly arrive at

$$\begin{aligned}
& {}_{\beta}\overset{a}{\longleftarrow} \cdot S \subseteq S \cdot {}_{\alpha}\overset{a}{\longleftarrow} \\
\equiv \quad & \{ \text{ Galois connection: } (R\cdot) \dashv (R\backslash) \; \} \\
& S \subseteq {}_{\beta}\overset{a}{\longleftarrow} \backslash (S \cdot {}_{\alpha}\overset{a}{\longleftarrow}) \\
\equiv \quad & \{ \text{ going pointwise } \} \\
& \forall_{v\in V, u\in U} \,.\; vSu \Rightarrow v(\,{}_{\beta}\overset{a}{\longleftarrow} \backslash (S \cdot {}_{\alpha}\overset{a}{\longleftarrow}))u \\
\equiv \quad & \{ \text{ pointwise definition of relational right division } \backslash \; \} \\
& \forall_{v\in V, u\in U} \,.\; vSu \Rightarrow (\forall_{v'\in V} \,.\; v' \,{}_{\alpha}\overset{a}{\longleftarrow}\, v \Rightarrow v'(\,{}_{\beta}\overset{a}{\longleftarrow} \cdot S)u') \\
\equiv \quad & \{ \text{ pointwise definition of relational composition } \cdot \} \\
& \forall_{v\in V, u\in U} \,.\; vSu \Rightarrow (\forall_{v'\in V} \,.\; v' \,{}_{\alpha}\overset{a}{\longleftarrow}\, v \Rightarrow (\exists_{u'\in U} \,.\; u' \,{}_{\beta}\overset{a}{\longleftarrow}\, u \wedge v'Su'))
\end{aligned}$$

which, in conjunction with (14), is the well-known expression used to define bisimulation in classical process algebra (*cf.*, [18,15]).

A useful result whose proof becomes almost trivial using formulation (16) is that the existence of a coalgebra morphism between two coalgebras relates by a bisimulation the pairs of states it connects. Formally,

Lemma 3.6 *The graph of a morphism $h : \beta \longleftarrow \alpha$ between coalgebras α and β is a bisimulation.*

Proof. As a coalgebra morphism connecting α and β, h verifies inequations (10) and (11). The latter is equivalent to the second conjunct in (16). A similar correspondence holds between the first conjunct and (10):

$$\begin{aligned}
& h \cdot {}_{\alpha}\overset{a}{\longleftarrow} \subseteq {}_{\beta}\overset{a}{\longleftarrow} \cdot h \\
\equiv \quad & \{ \text{ law (4) } \} \\
& {}_{\alpha}\overset{a}{\longleftarrow} \subseteq h^{\circ} \cdot {}_{\beta}\overset{a}{\longleftarrow} \cdot h \\
\equiv \quad & \{ \text{ converse is monotonic } \} \\
& (\,{}_{\alpha}\overset{a}{\longleftarrow})^{\circ} \subseteq (h^{\circ} \cdot {}_{\beta}\overset{a}{\longleftarrow} \cdot h)^{\circ} \\
\equiv \quad & \{ \text{ converse definition } \} \\
& \overset{a}{\longrightarrow}_{\alpha} \subseteq h^{\circ} \cdot \overset{a}{\longrightarrow}_{\beta} \cdot h \\
\equiv \quad & \{ \text{ law (4) } \} \\
& h \cdot \overset{a}{\longrightarrow}_{\alpha} \subseteq \overset{a}{\longrightarrow}_{\beta} \cdot h
\end{aligned}$$

□

4 Weak Bisimulation

4.1 Observational Reduction and Weak Bisimulations

Weak notions of equivalence, which abstract away specific subsets of actions considered *internal* or *non observable*, have a fundamental role in process calculi. However they are difficult to capture directly in a coalgebraic setting (but see [20], mentioned in section 5). In this section we approach a particular instance of the problem based on relational transposition. The starting point is the classical definition of weak bisimulation wrt a set $\Upsilon \subseteq A$ encoding internal actions. In Ccs, for example, $\Upsilon = \{\tau\}$.

Definition 4.1 Given ${}_{\alpha}\!\longleftarrow : A \times U \longleftarrow U$ e ${}_{\beta}\!\longleftarrow : A \times V \longleftarrow V$ over A and a subset $\Upsilon \subseteq A$ of non observable actions, a *weak simulation* of ${}_{\alpha}\!\longleftarrow$ in ${}_{\beta}\!\longleftarrow$ is a relation $S : V \longleftarrow U$ such that

$$\begin{aligned} \forall_{u \in U, v \in V} \,.\, vSu \;\Rightarrow\; & \forall_{a \in A - \Upsilon} \forall_{u' \in U} \,.\, u' \;{}_{\alpha}\!\overset{a}{\Longleftarrow}\; u \Rightarrow (\exists_{v' \in V} \,.\, v' \;{}_{\beta}\!\overset{a}{\Longleftarrow}\; v \wedge v' S u') \;\wedge \\ & \forall_{u' \in U} \,.\, u' \;{}_{\alpha}\!\Longleftarrow\; u \Rightarrow (\exists_{v' \in V} \,.\, v' \;{}_{\beta}\!\Longleftarrow\; v \wedge v' S u') \end{aligned}$$

where ${}_{\alpha}\!\Longleftarrow$ is the union, for all $\tau \in \Upsilon$, of the transitive, reflexive closure of ${}_{\alpha}\!\overset{\tau}{\longleftarrow}$, denoted by tr$({}_{\alpha}\!\overset{\tau}{\longleftarrow})$. Relation ${}_{\alpha}\!\overset{a}{\Longleftarrow}$ is defined by abbreviation:

$$ {}_{\alpha}\!\overset{a}{\Longleftarrow} \;\overset{\text{abv}}{=}\; {}_{\alpha}\!\Longleftarrow \cdot {}_{\alpha}\!\overset{a}{\longleftarrow} \cdot {}_{\alpha}\!\Longleftarrow \tag{17}$$

for all $a \in A - \Upsilon$. A *weak bisimulation* is a weak simulation whose converse is still a weak simulation.

Clearly the union of all weak bisimulations, denoted by $\approx$, is a weak bisimulation and an equivalence relation. Our strategy consists of transforming each coalgebra α into another coalgebra $\widehat{\alpha}$ such that a strict bisimulation over $\widehat{\alpha}$ will correspond to a weak one over the original α. Therefore, the standard procedure which seeks for a morphism to witness a bisimulation, remains valid once applied to $\widehat{\alpha}$, instead of α. The construction of $\widehat{\alpha}$ is depicted in the following diagram:

$$\begin{tikzcd} \alpha : \mathcal{P}(A \times U) \longleftarrow U \arrow[r, "\in \cdot"] & {}_{\alpha}\!\longleftarrow : A \times U \longleftarrow U \arrow[d, "\mathbb{O}"] \\ \widehat{\alpha} : \mathcal{P}(A \times U) \longleftarrow U & {}_{\alpha}\!\leftsquigarrow : A \times U \longleftarrow U \arrow[l, "\Lambda"'] \end{tikzcd}$$

Formally,

Definition 4.2 The *observational reduction* $\widehat{\alpha}$ of a coalgebra $\alpha : \mathcal{P}(A \times U) \longleftarrow U$, is defined by

$$\widehat{\alpha} \;=\; \Lambda\,\mathbb{O}(\in \cdot \alpha) \;=\; \Lambda\,\mathbb{O}({}_{\alpha}\!\longleftarrow) \;=\; \Lambda \;{}_{\alpha}\!\leftsquigarrow \tag{18}$$

where ${}_{\alpha}\!\leftsquigarrow$

$$(a, u') \;{}_{\alpha}\!\leftsquigarrow\; u \;\equiv\; u' \;{}_{\alpha}\!\overset{a}{\Longleftarrow}\; u \quad (\text{if } a \notin \Upsilon) \tag{19}$$

$$(\tau, u') \;{}_{\alpha}\!\leftsquigarrow\; u \;\equiv\; u' \;{}_{\alpha}\!\Longleftarrow\; u \quad (\text{if } \tau \in \Upsilon) \tag{20}$$

Clearly, a weak bisimulation over two coalgebras coincides with a strict bisimulation between the corresponding observational reductions. By construction an observational reduction of γ encodes all of its ${}_{\gamma}\!\overset{a}{\Longleftarrow}$ and ${}_{\gamma}\!\Longleftarrow$ transitions. Therefore a morphism between them, which preserves and reflects transitions, entails a weak bisimulation between the underlying coalgebras. Such a morphism is a coalgebra morphism, as shown in the previous section. Then, by lemma 3.6, it is a witness of a strict bisimulation between the corresponding observational reductions.

Example 4.3 Consider coalgebras α and β corresponding to transition systems:

$$\alpha \; = \; \begin{tikzcd} s_1 \arrow[r, "a"] \arrow[dr, "i"'] & s_2 & \\ & s_3 \arrow[r, "a"] & s_4 \arrow[loop right, "j"] \end{tikzcd} \qquad \beta \; = \; \begin{tikzcd} t_1 \arrow[r, "a"] & t_2 \end{tikzcd}$$

For $\Upsilon = \{i, j\}$ their *observable reductions* are

$$\widehat{\alpha} \; = \; \begin{tikzcd} s_1 \arrow[loop left, dotted, "\tau"] \arrow[r, "a"] \arrow[dr, "\tau"'] \arrow[drr, bend left, dotted, "a"] & s_2 \arrow[loop right, dotted, "\tau"] & \\ & s_3 \arrow[r, "a"] \arrow[loop below, dotted, "\tau"] & s_4 \arrow[loop right, dotted, "\tau"] \end{tikzcd} \qquad \widehat{\beta} \; = \; \begin{tikzcd} t_1 \arrow[loop left, dotted, "\tau"] \arrow[r, "a"] & t_2 \arrow[loop right, dotted, "\tau"] \end{tikzcd}$$

Define a morphism $h : \{t_1, t_2\} \longleftarrow \{s_1, s_2, s_3, s_4\}$ connecting states s_1 and t_1, initial in α and β, respectively, as follows: $h\, s_1 \; = \; h\, s_3 \; = \; t_1$ and $h\, s_2 \; = \; h\, s_4 \; = \; t_2$. The reader may easily check that $\widehat{\beta} \cdot h \; = \; \mathcal{P}(\mathsf{id} \times h) \cdot \widehat{\alpha}$.

Example 4.4 As a second example, consider CCS processes $c \cdot P$ and $\tau \cdot \tau \cdot c \cdot P$, with τ standing for internal activity. The denotation of each process term P in the final coalgebra carrier is $[P] \in \nu$. The relevant fragments of ω are depicted as follows:

$$\begin{tikzcd} {[c \cdot \mathsf{nil}]} \arrow[d, "c"'] & {[\tau \cdot c \cdot \mathsf{nil}]} \arrow[l, "\tau"'] & {[\tau \cdot \tau \cdot c \cdot \mathsf{nil}]} \arrow[l, "\tau"'] & {[c \cdot \mathsf{nil}]} \arrow[d, "c"] \\ {[\mathsf{nil}]} & & & {[\mathsf{nil}]} \end{tikzcd}$$

The corresponding fragments in the observational reduction $\widehat{\omega}$ of final coalgebra ω are:

$$\begin{tikzcd} {[c \cdot \mathsf{nil}]} \arrow[loop above, dotted, "\tau"] \arrow[d, "c"'] & {[\tau \cdot c \cdot \mathsf{nil}]} \arrow[loop above, dotted, "\tau"] \arrow[l, "\tau"'] \arrow[dl, dotted, "c"] & {[\tau \cdot \tau \cdot c \cdot \mathsf{nil}]} \arrow[loop above, dotted, "\tau"] \arrow[l, "\tau"'] \arrow[ll, bend right=40, dotted, "\tau"'] \arrow[dll, bend left, dotted, "c"] & {[c \cdot \mathsf{nil}]} \arrow[loop above, dotted, "\tau"] \arrow[d, "c"] \\ {[P]} \arrow[loop below, dotted, "\tau"] & & & {[\mathsf{nil}]} \arrow[loop below, dotted, "\tau"] \end{tikzcd}$$

Morphism $h : \widehat{\omega} \longleftarrow \widehat{\omega}$ defined by $h[\tau \cdot \tau \cdot c \cdot \mathsf{nil}] \; = \; h[\tau \cdot c \cdot \mathsf{nil}] \; = \; [c \cdot \mathsf{nil}]$ and as the identity in all other elements of ν, establishes a strict bisimulation over $\widehat{\omega}$ between the denotations of $c \cdot \mathsf{nil}$ and $\tau \cdot \tau \cdot c \cdot \mathsf{nil}$. Therefore, $c \cdot \mathsf{nil} \;\approx\; \tau \cdot \tau \cdot c \cdot \mathsf{nil}$.

The following lemma establishes that strict bisimilarity is contained in $\approx$.

Lemma 4.5 *Any morphism h from $\alpha : \mathcal{P}(Act \times U) \longleftarrow U$ to $\beta : \mathcal{P}(Act \times V) \longleftarrow V$ is also a morphism from $\widehat{\alpha}$ to $\widehat{\beta}$.*

Proof. Let h be such that $\beta \cdot h = \mathcal{P}(\mathsf{id} \times h) \cdot \alpha$. We show that $\widehat{\beta} \cdot h = \mathcal{P}(\mathsf{id} \times h) \cdot \widehat{\alpha}$ as follows

$$
\begin{array}{ll}
& \widehat{\beta} \cdot h \;=\; \mathcal{P}(\mathsf{id} \times h) \cdot \widehat{\alpha} \\
\equiv & \quad \{ \text{ definition 4.2 } \} \\
& \Lambda \cdot {}_{\beta}\!\leftsquigarrow \cdot h \;=\; \mathcal{P}(\mathsf{id} \times h) \cdot \Lambda \cdot {}_{\alpha}\!\leftsquigarrow \\
\equiv & \quad \{ \; \Lambda(f \cdot R) = \mathcal{P} f \cdot \Lambda R \; \} \\
& \Lambda \cdot {}_{\beta}\!\leftsquigarrow \cdot h \;=\; \Lambda \cdot (\mathsf{id} \times h) \cdot {}_{\alpha}\!\leftsquigarrow \\
\equiv & \quad \{ \; \Lambda \text{ is an isomorphism } \} \\
& {}_{\beta}\!\leftsquigarrow \cdot h \;=\; (\mathsf{id} \times h) \cdot {}_{\alpha}\!\leftsquigarrow
\end{array}
$$

To prove this last equality, let $a \notin \Upsilon$ and check that ${}_{\beta}\!\overset{a}{\leftsquigarrow} \cdot h \;=\; h \cdot {}_{\alpha}\!\overset{a}{\leftsquigarrow}$.

$$
\begin{array}{ll}
& {}_{\beta}\!\overset{a}{\leftsquigarrow} \cdot h \\
\equiv & \quad \{ \text{ definition of } {}_{\beta}\!\leftsquigarrow \} \\
& {}_{\beta}\!\Longleftarrow \cdot {}_{\beta}\!\overset{a}{\longleftarrow} \cdot {}_{\beta}\!\Longleftarrow \cdot h \\
\equiv & \quad \{ \; {}_{\beta}\!\Longleftarrow = \mathsf{tr}({}_{\beta}\!\overset{\tau}{\longleftarrow}) \; \} \\
& \mathsf{tr}({}_{\beta}\!\overset{\tau}{\longleftarrow}) \cdot {}_{\beta}\!\overset{a}{\longleftarrow} \cdot \mathsf{tr}({}_{\beta}\!\overset{\tau}{\longleftarrow}) \cdot h \\
\equiv & \quad \{ \; h : \beta \longleftarrow \alpha \text{ is a morphism } \} \\
& h \cdot \mathsf{tr}({}_{\alpha}\!\overset{\tau}{\longleftarrow}) \cdot {}_{\alpha}\!\overset{a}{\longleftarrow} \cdot \mathsf{tr}({}_{\alpha}\!\overset{\tau}{\longleftarrow}) \\
\equiv & \quad \{ \; {}_{\alpha}\!\Longleftarrow = \mathsf{tr}({}_{\alpha}\!\overset{\tau}{\longleftarrow}) \; \} \\
& h \cdot {}_{\alpha}\!\Longleftarrow \cdot {}_{\alpha}\!\overset{a}{\longleftarrow} \cdot {}_{\alpha}\!\Longleftarrow \\
\equiv & \quad \{ \text{ definition of } {}_{\beta}\!\leftsquigarrow \} \\
& h \cdot {}_{\alpha}\!\overset{a}{\leftsquigarrow}
\end{array}
$$

□

4.2 Proving ≈-laws

Equipped with a suitable notion of weak bisimulation, which is moreover parametric on a set of internal actions Υ, we may come back to our original motivation: rephrasing process algebras in a coalgebraic setting and reasoning coinductively in an equational style. The couple of examples in this section illustrate the kind of results we have in mind and the corresponding proof strategy.

Example 4.6 The first example is the law which in, CCS, characterizes observational equivalence:

$$p \;\approx\; \tau \cdot p \tag{21}$$

To show this we seek for a suitable morphism $h : \widehat{\omega} \longleftarrow \widehat{\omega}$. Let then

$$\begin{aligned} h(\tau \cdot p) &= p \quad \text{for all } p \in \nu \\ h &= \mathsf{id} \quad \text{otherwise} \end{aligned}$$

It remains to show that h, as defined, is actually a morphism, *i.e.*, $\widehat{\omega} \cdot h \;=\; \mathcal{P}(\mathsf{id} \times h) \cdot \widehat{\omega}$ which, given the definition of h, reduces to $\widehat{\omega}(\tau \cdot p) \;=\; \widehat{\omega} p$.

$$\begin{aligned}
& \widehat{\omega}(\tau \cdot p) \\
\equiv \quad & \{ \text{ definition 4.2 } \} \\
& \Lambda \;{}_{\omega}\!\leftsquigarrow\; (\tau \cdot p) \\
\equiv \quad & \{ \text{ definition of } {}_{\omega}\!\leftsquigarrow \text{ and transposition } \} \\
& \{(a, p') |\; p' \;{}_{\omega}\!\overset{a}{\Longleftarrow}\; \tau \cdot p\} \;\cup\; \{(\tau, p') |\; p' \;{}_{\omega}\!\Longleftarrow\; \tau \cdot p\} \\
\equiv \quad & \{ \text{ definition of } {}_{\omega}\!\Longleftarrow \} \\
& \{(a, p') |\; p' \;{}_{\omega}\!\overset{a}{\Longleftarrow}\; p\} \;\cup\; \{(\tau, p') |\; p' \;{}_{\omega}\!\Longleftarrow\; p\} \\
\equiv \quad & \{ \text{ definition of } {}_{\omega}\!\leftsquigarrow \text{ and transposition } \} \\
& \Lambda \;{}_{\omega}\!\leftsquigarrow\; p \\
\equiv \quad & \{ \text{ definition 4.2 } \} \\
& \widehat{\omega}\, p
\end{aligned}$$

Example 4.7 Consider, now, the following conditional law:

$$p + q \;\approx\; p' + q' \quad \Leftarrow \quad p' \approx p \;\wedge\; q' \approx q \tag{22}$$

Again we seek for a morphism h connecting the relevant terms such that

$$\widehat{\omega} \cdot + \cdot (h \times h) \;=\; \mathcal{P}(\mathsf{id} \times h) \cdot \widehat{\omega} \cdot + \tag{23}$$

This equation requires the definition of combinator + over $\widehat{\omega}$. Recall that $\omega \cdot + \;= \cup \cdot (\omega \times \omega)$, which, once combined with the definition of observable reduction in (18), leads to

$$\widehat{\omega}([p + q]) \;=\; \widehat{\omega} p \cup \widehat{\omega} q \;\cup\; \{(\tau, r) |\; r \;{}_{\omega}\!\Longleftarrow\; [p + q]\} \tag{24}$$

Thus $\widehat{\omega}([a \cdot \mathsf{nil} \;+\; b \cdot \mathsf{nil}]) \;=\; \{(a, [\mathsf{nil}]), (b, [\mathsf{nil}]), (\tau, [(a \cdot \mathsf{nil} \;+\; b \cdot \mathsf{nil})])\}$ whereas $\widehat{\omega}([\tau \cdot a \cdot \mathsf{nil} \;+\; b \cdot \mathsf{nil}]) \;=\; \{(a, [\mathsf{nil}]), (b, [\mathsf{nil}]), (\tau, [(a \cdot \mathsf{nil} \;+\; b \cdot \mathsf{nil})]), (\tau, [a \cdot \mathsf{nil}])\}$ although, as shown in the previous example, $a \cdot \mathsf{nil} \approx \tau \cdot a \cdot \mathsf{nil}$. Therefore, equation (22) does not hold.

Example 4.8 Consider, finally, a congruence law for the interleaving combinator:

$$p \;|||\; q \;\approx\; p' \;|||\; q' \quad \Leftarrow \quad p' \approx p \;\wedge\; q' \approx q \tag{25}$$

Suppose equivalences $p' \approx p$ and $q' \approx q$ are witnessed by morphisms $f : \widehat{\omega} \longleftarrow \widehat{\omega}$ and $g : \widehat{\omega} \longleftarrow \widehat{\omega}$, not necessarily coincidents. Define $h : \nu \longleftarrow \nu$ such that

$$h \cdot ||| \;=\; ||| \cdot (f \times g) \tag{26}$$

and is the identity in all other cases. Let us show that h is a morphism between the observational reductions of ω, *i.e.*,

$$\widehat{\omega} \cdot h \;=\; \mathcal{P}(\mathsf{id} \times h) \cdot \widehat{\omega} \tag{27}$$

Clearly, equation (27) holds for all arguments for which h is the identity. Therefore, the only case to be checked is the application to interleaving expressions (*e.g.*, $p\,|||\,q$), reducing our task to prove

$$\widehat{\omega} \cdot h \cdot ||| \;=\; \mathcal{P}(\mathsf{id} \times h) \cdot \widehat{\omega} \cdot ||| \tag{28}$$

By definition of h, $\widehat{\omega} \cdot h \cdot |||\; = \;\widehat{\omega} \cdot ||| \cdot (f \times g)$. But what can be said about term $\widehat{\omega} \cdot |||$? Note that diagram

$$\begin{tikzcd}
\nu \arrow[r, "\omega"] & \mathcal{P}(Act \times \nu) \\
\nu \times \nu \arrow[u, "|||"] \arrow[r, "\alpha_{|||}"] & \mathcal{P}(Act \times (\nu \times \nu)) \arrow[u, "\mathcal{P}(\mathsf{id} \times |||)"']
\end{tikzcd}$$

generalizes to

$$\begin{tikzcd}
\nu \arrow[r, "\widehat{\omega}"] & \mathcal{P}(Act \times \nu) \\
\nu \times \nu \arrow[u, "|||"] \arrow[r, "\widehat{\alpha_{|||}}"] & \mathcal{P}(Act \times (\nu \times \nu)) \arrow[u, "\mathcal{P}(\mathsf{id} \times |||)"']
\end{tikzcd}$$

where $\widehat{\alpha_{|||}} \;=\; \Lambda\ {}_{\alpha_{|||}} \leftsquigarrow$, just as $\widehat{\omega} \;=\; \Lambda\ {}_{\omega} \leftsquigarrow$. Actually, by lemma 4.5, the latter diagram is implied by the former. It is not difficult to find a direct definition for $\Lambda\ {}_{\alpha_{|||}} \leftsquigarrow$: it is enough to replace ω by $\widehat{\omega}$ in the definition of $\alpha_{|||}$ given in the end of section 2, and consider a τ-labelled transition from the pair of arguments to itself. For example, $\widehat{\alpha_{|||}}\,(\tau \cdot a \cdot \mathsf{nil}, b \cdot \mathsf{nil})$ is the union of singleton $\{(\tau, ([\tau \cdot a \cdot \mathsf{nil}], [b \cdot \mathsf{nil}]))\}$ with $\{(\tau, ([a \cdot \mathsf{nil}], [b \cdot \mathsf{nil}])), (a, ([\mathsf{nil}], [b \cdot \mathsf{nil}])), (b, ([a \cdot \mathsf{nil}], [\mathsf{nil}]))\}$. Therefore,

$$\widehat{\alpha_{|||}} \;=\; \cup \cdot (\alpha_1 \times \alpha_2) \cdot \vartriangle \tag{29}$$

where $\alpha_1 \;= \cup \cdot\ (\tau_r \times \tau_l) \cdot ((\widehat{\omega} \times \mathsf{id}) \times (\mathsf{id} \times \widehat{\omega})) \cdot \vartriangle$ and $\alpha_2 \;=\; \mathsf{sing} \cdot \mathsf{label}_\tau$. Now we check, for $i = 1, 2$, that

$$\mathcal{P}(\mathsf{id} \times |||) \cdot \alpha_i \cdot (f \times g) \;=\; \mathcal{P}(\mathsf{id} \times h) \cdot \mathcal{P}(\mathsf{id} \times |||) \cdot \alpha_i \tag{30}$$

The case $i = 1$ is proved by

$$
\begin{aligned}
& \mathcal{P}(\mathsf{id} \times |||) \cdot \alpha_1 \cdot (f \times g) \\
= & \quad \{ \text{ definition of } \alpha_1 \ \} \\
& \mathcal{P}(\mathsf{id} \times |||) \cdot \cup \cdot (\tau_r \times \tau_l) \cdot ((\widehat{\omega} \times \mathsf{id}) \times (\mathsf{id} \times \widehat{\omega})) \cdot \vartriangle \cdot (f \times g) \\
= & \quad \{ \ \vartriangle \text{ natural, } \times \text{ functor } \} \\
& \mathcal{P}(\mathsf{id} \times |||) \cdot \cup \cdot (\tau_r \times \tau_l) \cdot (((\widehat{\omega} \cdot f) \times g) \times (f \times (\widehat{\omega} \cdot g))) \cdot \vartriangle \\
= & \quad \{ \ f \text{ and } g \text{ are morphisms , } \times \text{ functor } \} \\
& \mathcal{P}(\mathsf{id} \times |||) \cdot \cup \cdot (\tau_r \times \tau_l) \cdot (((\mathcal{P}(\mathsf{id} \times f) \times g) \times (f \times \mathcal{P}(\mathsf{id} \times g))) \cdot ((\widehat{\omega} \times \mathsf{id}) \times (\mathsf{id} \times \widehat{\omega})) \cdot \vartriangle \\
= & \quad \{ \ \tau_r, \tau_l \text{ and } \cup \text{ natural } \} \\
& \mathcal{P}(\mathsf{id} \times |||) \cdot \mathcal{P}(\mathsf{id} \times (f \times g)) \cdot \cup \cdot (\tau_r \times \tau_l) \cdot ((\widehat{\omega} \times \mathsf{id}) \times (\mathsf{id} \times \widehat{\omega})) \cdot \vartriangle \\
= & \quad \{ \text{ definition of } \alpha_1 \ \} \\
& \mathcal{P}(\mathsf{id} \times |||) \cdot \mathcal{P}(\mathsf{id} \times (f \times g)) \cdot \alpha_1 \\
= & \quad \{ \ (26) \ \} \\
& \mathcal{P}(\mathsf{id} \times h) \cdot \mathcal{P}(\mathsf{id} \times |||) \cdot \alpha_1
\end{aligned}
$$

The proof of case $i = 2$ is similar. Finally, equation (28) is checked as follows,

$$
\begin{aligned}
& \widehat{\omega} \cdot h \cdot ||| \\
= & \quad \{ \ (26) \ \} \\
& \widehat{\omega} \cdot ||| \cdot (f \times g) \\
= & \quad \{ \ ||| \text{ is a morphism for } \widehat{\omega} \ \} \\
& \mathcal{P}(\mathsf{id} \times |||) \cdot \widehat{\alpha_{|||}} \cdot (f \times g) \\
= & \quad \{ \text{ definition of } \widehat{\alpha_{|||}} \ \} \\
& \mathcal{P}(\mathsf{id} \times |||) \cdot \cup \cdot (\alpha_1 \times \alpha_2) \cdot \vartriangle \cdot (f \times g) \\
= & \quad \{ \ \vartriangle \text{ and } \cup \text{ natural, } \times \text{ functor } \} \\
& \cup \cdot ((\mathcal{P}(\mathsf{id} \times |||) \cdot \alpha_1 \cdot (f \times g)) \times (\mathcal{P}(\mathsf{id} \times |||) \cdot \alpha_2 \cdot (f \times g))) \cdot \vartriangle \\
= & \quad \{ \text{ both instances of equation } (30) \ \} \\
& \cup \cdot ((\mathcal{P}(\mathsf{id} \times h) \cdot \mathcal{P}(\mathsf{id} \times |||) \cdot \alpha_1) \times (\mathcal{P}(\mathsf{id} \times h) \cdot \mathcal{P}(\mathsf{id} \times |||) \cdot \alpha_2)) \cdot \vartriangle \\
= & \quad \{ \ \cup \text{ natural } \} \\
& \mathcal{P}(\mathsf{id} \times h) \cdot \mathcal{P}(\mathsf{id} \times |||) \cdot \cup \cdot (\alpha_1 \times \alpha_2) \cdot \vartriangle \\
= & \quad \{ \text{ definition of } \widehat{\alpha_{|||}} \ \} \\
& \mathcal{P}(\mathsf{id} \times h) \cdot \mathcal{P}(\mathsf{id} \times |||) \cdot \widehat{\alpha_{|||}} \\
= & \quad \{ \ ||| \text{ is a morphism for } \widehat{\omega} \ \} \\
& \mathcal{P}(\mathsf{id} \times h) \cdot \widehat{\omega} \cdot |||
\end{aligned}
$$

5 Conclusions and Future Work

Set as a simple exercise, this paper introduced a concrete notion of weak bisimulation for labelled transition systems, which is parametric on a definition of *internal activity*. As a 'by-product', the paper illustrated how basic results in transition systems theory admit intuitive and simple characterizations and proofs, when formulated in a pointfree style.

Lots of questions remain to be answered. The most proeminent asks to what extent this approach can be extended to broader classes of coalgebras. The quest for canonical ways of absracting from hidden transitions is still an open question in coalgebra research. Generalizing the approach sketeched here entails the need for new transposition operators possibly to categories different from **Rel**. A very simple case relates coalgebras for the *maybe* monad with a category of partial functions, but such is just a specialization of the case dealt in this paper. The interesting question would be to consider coalgebras expressing probabilistic behaviour or particular timing constraints, while retaining the simplicity and calculational flavour of our approach. Related work includes [19], which deals with **Set** functors for which a notion of *natural accessor* can be defined, and, more recently, [20]. The latter, rather generic although resorting to heavy (categorial) notation, is based on process traces factorised wrt a set of invisible actions. It should be stressed, however, that the motivation for this paper was somewhat different: we looked for a *concrete* notion of weak bisimulation to be used in effective and simple calculations with processes' denotations.

Another question concerns the possible scalling up of this work to process algebras with *mobility*. However, we are still far from developing a coinductive, calculational, account of the π-calculus [16] along the lines of [6], even if its (coalgebraic) semantics has already been tackled, at a foundational level, in *e.g.* [13,5]. We are currently working on this topic resorting to coalgebras over dependent types.

References

[1] P. Aczel, *Final universes of processes*, Proc. Math. Foundations of Programming Semantics (Brooks et al, ed.), Springer Lect. Notes Comp. Sci. (802), 1993.

[2] L. Alfaro, T. Henziger, and R. Jhala, *Compositional methods for probabilistic systems*, Proc. the 12th International Conference on Concurrency Theory, Springer Lect. Notes Comp. Sci. (2154), 2001, pp. 351–365.

[3] R. Backhouse, *Galois connections and fixed point calculus*, Algebraic and Coalgebraic Methods in the Mathematics of Program Constuction (R. Crole, R. Backhouse, and J. Gibbons, eds.), Springer Lect. Notes Comp. Sci. (2297), 2002, pp. 89–148.

[4] R. C. Backhouse and P. F. Hoogendijk, *Elements of a relational theory of datatypes*, Formal Program Development (B. Möller, H. Partsch, and S. Schuman, eds.), Springer Lect. Notes Comp. Sci. (755), 1993, pp. 7–42.

[5] M. Baldamus, *Compositional constructor interpretation over coalgebraic models for the π-calculus*, Proc. of CMCS'00 (H. Reichel, ed.), vol. 33, Elect. Notes in Theor. Comp. Sci., Elsevier, 2000.

[6] L. S. Barbosa, *Process calculi* à la *Bird-Meertens*, CMCS'01 (Genova), vol. 44.4, Elect. Notes in Theor. Comp. Sci., Elsevier, April 2001, pp. 47–66.

[7] L. S. Barbosa and J. N. Oliveira, *Coinductive interpreters for process calculi*, Proc. of FLOPS'02, Springer Lect. Notes Comp. Sci. (2441), September 2002, pp. 183–197.

[8] R. Bird and O. Moor, *The algebra of programming*, Series in Computer Science, Prentice-Hall International, 1997.

[9] R. S. Bird and L. Meertens, *Two exercises found in a book on algorithmics*, Program Specification and Transformation (L. Meertens, ed.), North-Holland, 1987, pp. 451–458.

[10] T. Hagino, *A typed lambda calculus with categorical type constructors*, Category Theory and Computer Science (D. H. Pitt, A. Poigné, and D. E. Rydeheard, eds.), Springer Lect. Notes Comp. Sci. (283), 1987, pp. 140–157.

[11] T. Henziger, *Hybrid automata with finite bisimulations*, Proc. the 22th Inter. Colloquium on Automata, Languages, and Programming (ICALP), Springer Lect. Notes Comp. Sci. (944), 1995, pp. 324–335.

[12] K. Larsen and A. Skou, *Bisimulation through probabilistic testing*, Information and Computation (1991), no. 94, 1–28.

[13] M. Lenisa, *Themes in final semantics*, Ph.D. thesis, Universita de Pisa-Udine, 1998.

[14] E. Meijer, M. Fokkinga, and R. Paterson, *Functional programming with bananas, lenses, envelopes and barbed wire*, Proceedings of the 1991 ACM Conference on Functional Programming Languages and Computer Architecture (J. Hughes, ed.), Springer Lect. Notes Comp. Sci. (523), 1991, pp. 124–144.

[15] R. Milner, *Communication and concurrency*, Series in Computer Science, Prentice-Hall International, 1989.

[16] R. Milner, J. Parrow, and D. Walker, *A calculus of mobile processes (parts I and II)*, Information and Computation **100** (1992), no. 1, 1–77.

[17] J. N. Oliveira and C. J. Rodrigues, *Transposing relations: From* Maybe *functions to hash tables*, 7th International Conference on Mathematics of Program Construction (D. Kozen, ed.), Springer Lect. Notes Comp. Sci. (3125), July 2004, pp. 334–356.

[18] D. Park, *Concurrency and automata on infinite sequences*, Springer Lect. Notes Comp. Sci. (104), 1981, pp. 561–572.

[19] J. Rothe and D. Masulovic, *Towards weak bisimulation for coalgebras*, Proc. Int. Workshop on Categorical Methods for Concurrency, Interaction, and Mobility (CMCIM'02) (A. Kurz, ed.), vol. 68, Elect. Notes in Theor. Comp. Sci., Elsevier, 2002.

[20] A. Sokolova, E. de Vink, and H. Woracek, *Weak bisimulation for action-type coalgebras*, Proc. Int. Conf. on Category Theory and Computer Science (CTCS'04) (L. Birkedal, ed.), vol. 122, Elect. Notes in Theor. Comp. Sci., Elsevier, 2005, pp. 211–228.