Labels from reductions: towards a general theory

Bartek Klin, Vladimiro Sassone, and Paweł Sobociński

Warsaw University, University of Sussex, and PPS – Université Paris VII

Abstract. We consider open terms and parametric rules in the context of the systematic derivation of labelled transitions from reduction systems.

1 Introduction

Since the seminal ideas of logicians of the early 20th century, it has become customary to encapsulate the dynamics of computation in terse and elegant *rewrite* calculi. For instance, the essence of conventional computation is condensed in Church's beta-reduction rule $(\lambda x.M)N \to M\{x := N\}$, while the mechanics of π-calculus interaction is captured by the rule

$$\overline{a}\langle n\rangle.P \mid a(x).Q \longrightarrow P \mid Q\{x := n\} \ .$$

The beauty and power of such formalisms can hardly be overestimated: they centre our models on the essential, and help us focus our reasoning on fundamental principles. However, models are normally used not only to describe, but also to design, specify, analyse, and – most importantly – as the foundations for advanced, ground-breaking techniques. A well consolidated, relevant example is 'model checking,' where simple tools used at a suitable abstraction level and driven by powerful ideas have afforded spectacular results. Similarly, notions revolving around semantic equivalences and coinduction have had a strong, lasting impact.

Several such ideas rely on relatively lower-level models based on *transition systems*. Intuitively, these describe individual steps of computing entities, rather than providing an overall picture of the computational primitives of the model as such. For instance, in the case of π-calculus terms a transition $\overline{a}\langle n\rangle.P \xrightarrow{\overline{a}n} P$ would express that the system is ready to evolve to P by engaging in action $\overline{a}$ and offering it to (potential partners in) the environment. There would then be a dual rule $a(x).Q \xrightarrow{a(n)} Q\{x := n\}$ for message receivers, and finally an inference rule would dictate how dual actions can meet in the environment and complete each other to yield finished interactions.

$$\frac{A \xrightarrow{\overline{a}n} A' \qquad B \xrightarrow{a(n)} B'}{A \mid B \longrightarrow A' \mid B'}$$

Although the resulting term-transformation systems are equivalent, the differences between these approaches are significant, and are better not dismissed hastily by a simple 'matter-of-taste' argument. The fundamental point of a 'labelled-transition' semantics is that it is compositional: it explains the behaviour of complex systems by extrapolating it from the behaviour of their components. This is in sharp contrast with a 'reduction' semantics, where $\overline{a}\langle n\rangle.P$ and $a(x).Q$ are completely inert, have no meaning of their own. This distinction is of paramount importance for applications like model

J.L. Fiadeiro et al. (Eds.): CALCO 2005, LNCS 3629, pp.30–50, 2005.

checking and bisimulation, that rely on the information afforded by labels and transitions to analyse system components in isolation.

Influenced by Plotkin's successful 'structural operational semantics' [10], many formalisms in the seventies and eighties had been originally equipped exclusively with a labelled-transition semantics, including CCS and the π-calculus. In recent years however it has become increasingly important for complex computational models to have both a reduction semantics, to explain their mechanics in intuitive, self-justifying terms, and a labelled-transition semantics, to serve as basis for semantic analysis. In particular, several papers have been devoted to identify characterisations of reduction-based equivalences in terms of labels and bisimulations. This is the context of the present work: is it possible, and how, to *derive labelled transition systems from reductions* so as to equip calculi with rich and treatable semantics theories? And to what extent can this be done parametrically, i.e., independently on the specific calculus at hand? Questions like these gained momentum as work on 'universal' models emerged from the field of concurrency, as e.g. action calculi [7], tile systems [2], and, more recently, bigraphs [3, 8]. Such models are meant to provide general frameworks independent of specific models, such that several calculi can be recast and understood as fragments therein. In ambitious terms, one could think of these frameworks as semantic universes which individual models can be instantiated from. The question therefore arose as to how to associate meaning and reasoning techniques to such 'universal' meta-models.

Much progress has been made since, mainly by Robin Milner and his collaborators. The rest of this introduction will revisit the main ideas underlying the approach, whilst the main body of paper will present the technical details in a slightly novel fashion, and try to accommodate in the theory the idea of parametric rules and open terms.

The central technical challenge is thus how to associate labelled transitions to terms from reduction systems. Peter Sewell [16] exploited the intuition that labels in labelled transition systems express the compositional properties of terms, i.e., the extend to which a term is amenable to engage in interactions with the environment, and how. Thus, if term a when inserted in a *context* $c[-]$ can perform a reduction, say $c[a] \to a'$, then $c[-]$ is a strong candidate as a label for a transition $a \xrightarrow{c} a'$. (This spells out as: 'a is ready to interact with context $c[-]$, and a' would be the result of such potential interaction.') This intuition is very suggestive indeed; the devil however is as usual the details: in order for this idea to give a sensible bisimulation, it is fundamental to select carefully which contexts to consider: certainly not all, but only those c which are the '*smallest*' to trigger a given reduction. Failing to do so would give rise to a 'garbled' semantics, as the excess transitions would convey misleading information as to what term a is ready to engage with and what the environment is expected to contribute.

The need to formalise the notion of 'smallest' leads to *category theory*, where it is possible to express such universal properties in term of uniqueness of certain 'arrow' factorisation. For instance, in categorical terms the fact that C is the disjoint union (the so-called coproduct) of sets A and B is expressed by saying that all pairs of maps (arrows) $f : A \to X$ and $g : B \to X$ factor uniquely via injections into C and a map $[f, g] : C \to X$. In complete analogy, a context $c[-]$ is the 'smallest' to create redex l in a, if all contexts $c'[-]$ that create l factor as $c'[-] = e[c[-]]$ for a unique context $e[-]$.

For instance, in the λ-calculus

$$\lambda x.x \xrightarrow{(-)y} y, \qquad \text{but not} \qquad \lambda x.x \xrightarrow{(-)yz} yz,$$

as $(-)yz$ arises uniquely as the composition of $(-)z$ and $(-)y$.

The first step to rephrase our notion of 'smallness' as a problem of unique arrow factorisation is to recast terms as arrows in categories. This can be done following Lawvere's seminal approach to algebraic theories, that here we instantiate using MacLane's notion of 'product and permutation' category (PROP) – roughly speaking, 'linear' Lawvere theories – that we recall in §2. An arrow $f : n \to m$ in a PROP represents a m-tuple of contexts containing altogether n 'holes;' i.e., when f is fed with n terms to plug its holes, it yields a tuple of m terms. The question as to whether or not term a in context c manifests a redex l becomes now whether there exists a suitable context d such that $ca = dl$. This allows us to express the minimality of c by ranging over all equations of the kind $c'a = d'l$, seeking for unique ways to factor c' through c. In §3 we recall how such universal property is elegantly expressed by the notion of *idem-relative-pushout*, a breakthrough due Leifer and Milner [4]. Remarkably, such formalisation supports the central '*congruence theorem*' that bisimulation on the labelled transition systems derived following the theory is a congruence, i.e., it is closed under all contexts. Due to the generality of the framework, such a result has already been applied to a variety of different models [1, 3, 9, 12–14]

This paper's original contribution concerns our initial ideas on the treatment of *open terms* and *parametric rules* in the above framework.

For the sake of illustration, let us consider on the CCS rule for interaction. To express such a rule as a collection of ground rules $\overline{a}P \mid aQ \longrightarrow P \mid Q$ is not entirely satisfactory in this setting: even in the simplest cases, we have infinitely many rewrite rules to deal with, and these give rise to infinitely many higher-order labels, e.g., of the kind $\overline{a}P \xrightarrow{-|aQ} P \mid Q$. This appears to make a poor use of the generality, elegance and succinctness of theory of relative pushouts. Ideally, the rule should be expressed parametrically, as in[1]

$$a.1 \mid \overline{a}.2 \longrightarrow 1 \mid 2$$

and the labels should be derivable for open terms with universal property imposed both on the contexts and the parameters. A label should thus consist both of a smallest context and the most general parameter which makes a reduction possible; for example

$$\langle a.P, 1\rangle \xrightarrow{1|\overline{a}.2} P \mid 1 \qquad \text{and} \qquad a.P \mid 1 \xrightarrow[\overline{a}.1]{} P \mid 1,$$

where the label above the transition denotes a context with two holes (to insert the left-hand pair in), and the label below a transition denotes a parameter (to fill the left-hand side open term with).

As it turns out, the framework is robust enough to adapt easily to the new question. Rather than investigating ('square') equations such as $ca = dl$, we now face '*hexagonal*' equations $cap = dlq$ in order to establish the universal property that, at the same time,

[1] In the paper we use natural numbers to denote context parameters ('holes').

identifies the smallest context c as well as the largest parameter p that unearth redex l in term a. The main technical device we introduce to that purpose is to pair the notion of slice pushout (a rephrasing of relative pushouts) with a dual notion of coslice pullback: the role of the pushout is to determine c as before, while the pullback of course ascertains p. Such coupling of universal properties gives rise to the new notion of '*lux*' (locally universal hexagon), introduced in §4. These have been considered previously by Peter Sewell, who referred to them as hex-RPOs. In fact, much of our technical development has been foreshadowed in his unpublished notes [15].

Our main technical results are a characterisation of categories with luxes in terms of slice pushouts and coslice pullbacks (Theorem 1) and, of course, the fundamental *congruence theorem* for the labelled transition systems derived using our theory of luxes (Theorem 3). Most of the ideas presented here are work in progress, and in the concluding section we discuss merits and shortcomings of our proposal, as well as identifying some of the main avenues of future work on luxes.

Structure of the paper. In §2 we recall the notion of PROP and the construction of categories of terms. §3 illustrates the existing theory based on slice pushouts, and its extension to a bicategorical setting. §4-6 contain the main body of the paper, with our definition of luxes, their properties, and the congruence theorem. Finally, §7 discusses the shortcomings of the current theory and points forward to open issues and future research.

We assume the reader to have a basic knowledge of category theory, as can be acquired from any graduate textbook. Throughout the paper we use standard categorical notations, where $\circ$ denotes (right-to-left) composition and is most often omitted.

2 PROPs as categories of terms

A 'product and permutation' category [5], PROP, can be described, roughly, as a linear Lawvere theory; more accurately, PROPs are one-sorted symmetric monoidal theories whereas Lawvere theories are one-sorted finite product theories. We recall a straightforward definition below.

Definition 1 (PROP). A PROP is a category $\mathbf{C}$ where:

- objects are the natural numbers (here denoted $\underline{0}, \underline{1}, \underline{2}, \ldots$);
- for each n, the group of permutations of n elements, $S(n)$, is a subgroup of all the invertible elements of the homset $[\underline{n}, \underline{n}]$. The identity permutation corresponds to the identity $1_{\underline{n}} : \underline{n} \to \underline{n}$;
- there is a functor $\otimes : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$ which acts as addition on the objects, i.e., $\underline{m} \otimes \underline{n} = \underline{m+n}$, and additionally:
  - is associative: $(f \otimes f') \otimes f'' = f \otimes (f' \otimes f'')$;
  - given $\sigma \in S(n)$ and $\sigma' \in S(n')$, we have $\sigma \otimes \sigma' = \sigma \times \sigma' : \underline{n+n'} \to \underline{n+n'}$, where $\times$ denotes the product of permutations;
  - for any two natural numbers n, n', let $\gamma_{\underline{n},\underline{n'}} : \underline{n+n'} \to \underline{n+n'}$ be the permutation which swaps the two blocks of n and n'. Then for any maps $f : \underline{m} \to \underline{n}$ and $f' : \underline{m'} \to \underline{n'}$ we have $\gamma_{\underline{n},\underline{n'}}(f \otimes f') = (f' \otimes f)\gamma_{\underline{m},\underline{m'}}$.

Example 1. For any algebraic signature (i.e., set of operator names with finite arities) Σ, the *free PROP* $\mathbf{P}_\Sigma$ over Σ has n-tuples of terms over Σ that altogether contain m distinct holes, as arrows $t : \underline{m} \to \underline{n}$. Permutations in $[\underline{n}, \underline{n}]$ are tuples built solely of holes, $\otimes$ acts on arrows as tuple juxtaposition, and arrow composition is the standard composition of terms.

Example 2. PROPs can also be induced from signatures modulo term equations. Consider the signature $\Sigma = \{\mathtt{nil} : 0,\ a. : 1,\ \overline{a}. : 1,\ | : 2\}$ corresponding to the grammar:

$$P ::= \mathtt{nil} \mid aP \mid \overline{a}P \mid P \mid P\ ,$$

where a ranges over some fixed set A of actions, and the associativity equation:

$$P \mid (Q \mid R) = (P \mid Q) \mid R\ .$$

The PROP **PAP** (Prefix and Associative Parallel composition) is built of terms over Σ quotiented by the associativity equation, with permutations, $\otimes$ and composition defined as in Example 1. Additionally, one can quotient terms by the commutativity equation

$$P \mid Q = Q \mid P\ ;$$

the resulting PROP will be called **PACP** (Prefix and Associative, Commutative Parallel Composition).

3 Labelled transitions for ground reductions

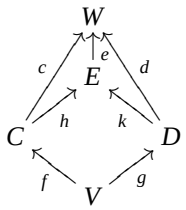
This section introduces the background material we need in later sections. First, we briefly recall Leifer and Milner's notion of idem-relative-pushout (IPO) as well as its dual, the idem-relative-pullback (IPB). Following a brief informal and discussion on how IPOs have been used in order to generate labelled transition systems (LTS) for calculi with ground reduction rules, we shall demonstrate that IPOs and IPBs can be conveniently studied in a category of factorisations, where they are easily seen to be coproducts and products, respectively. We conclude with a short note on how to generalise the theory to G-categories [11, 13]

3.1 Pushouts in slices

Let $\mathbf{C}$ be a category and V, W objects of $\mathbf{C}$. The *slice category* $\mathbf{C}/W$ has as objects pairs $\langle X, a\rangle$, where $a : X \to W$ is an arrow of $\mathbf{C}$, while its arrows $f\colon \langle X, a\rangle \to \langle X', a'\rangle$ are arrows $f\colon X \to X'$ in $\mathbf{C}$ such that $a' f = a$. The dual notion of a *coslice category* $V/\mathbf{C}$ consists of the pairs $\langle b, X\rangle$, where $b : V \to X$ and of maps $f\colon \langle b, X\rangle \to \langle b', X'\rangle$ for $f\colon X \to X'$ such that $fb = b'$.

Let $r\colon V \to W$ be an arrow of $\mathbf{C}$. A *pushout* of $f\colon \langle V, r\rangle \to \langle C, c\rangle$ and $g\colon \langle V, r\rangle \to \langle D, d\rangle$ in the slice category $\mathbf{C}/W$ is, equivalently, a *coproduct* in $\langle V, r\rangle /(\mathbf{C}/W)$. Spelling this definition out, the span of f and g identifies a commutative square $cf = r = dg$ in $\mathbf{C}$, while the pushout diagram $h\colon \langle C, c\rangle \to \langle E, e\rangle$ and $k\colon \langle D, d\rangle \to \langle E, e\rangle$ determines a

universal set of arrows such that $hf = kg$, $eh = c$ and $ek = d$, as in the diagram below. We shall say that a category has *slice pushouts* when it has pushouts in all slices.[2]



Lemma 1. *Free PROPs have slice pushouts.*

Proof (sketch). A diagram

$$\left\langle \underline{k}, c \right\rangle \xleftarrow{a} \left\langle \underline{m}, t \right\rangle \xrightarrow{b} \left\langle \underline{l}, d \right\rangle$$

in $\mathbf{P}_\Sigma/\underline{n}$ is an arrow $t : \underline{m} \to \underline{n}$ in $\mathbf{P}_\Sigma$ with its two decompositions:

$$ca = t = db$$

As usual, a *position* ρ in a given term t is a finite sequence of numbers which encodes a path downward from a root node of t. The set of positions in t, with the standard prefix ordering, is denoted S_t.

It is straightforward to check that decompositions of a given arrow t into p arrows are in 1-1 correspondence to monotonic functions from S_t to the set $\{1, \ldots, p\}$ with the natural ordering. Consider such functions Λ_{ca} and Λ_{db} corresponding to the two decompositions above, and define

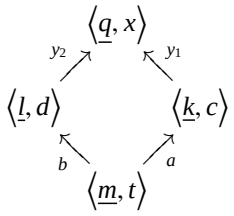
$$\Lambda_1(\rho) = \begin{cases} 1 \text{ if } \Lambda_{ca}(\rho) = 1 \text{ and } \Lambda_{db}(\rho) = 1 \\ 2 \text{ if } \Lambda_{ca}(\rho) = 1 \text{ and } \Lambda_{db}(\rho) = 2 \\ 3 \text{ if } \Lambda_{ca}(\rho) = 2 \end{cases}$$

$$\Lambda_2(\rho) = \begin{cases} 1 \text{ if } \Lambda_{ca}(\rho) = 1 \text{ and } \Lambda_{db}(\rho) = 1 \\ 2 \text{ if } \Lambda_{ca}(\rho) = 2 \text{ and } \Lambda_{db}(\rho) = 1 \\ 3 \text{ if } \Lambda_{db}(\rho) = 2 \end{cases}$$

Λ_1 and Λ_2 are monotonic, hence they correspond to two decompositions of t:

$$x_1 y_1 z_1 = t = x_2 y_2 z_2$$

Moreover, $x_1 = x_2$, $z_1 = a$ and $z_2 = b$. Let the domain of $x = x_1 = x_2$ be $\underline{q}$. The square



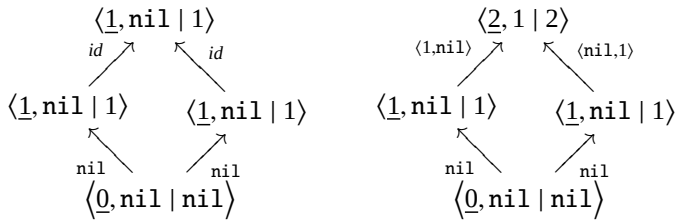
[2] Leifer and Milner [4] use the term relative pushouts, or RPOs, to refer to pushouts in slices.

is a pushout in $\mathbf{P}_\Sigma/\underline{n}$.

Lemma 2. ***PAP*** *has slice pushouts.*

Proof (sketch). Arrows in **PAP** can be represented as tuples of finite, ordered trees with nodes of any degree, where an immediate child of a node of degree higher than 1 must have degree at most 1. Additionally, nodes of degree 1 are labelled with elements of A. Leaves of such trees correspond to occurrences of the constant `nil`, nodes of degree 1 to applications of prefix composition operators, and nodes of higher degree to term fragments built solely of the associative parallel composition operator. On this representation of arrows, a pushout construction very similar to that of Lemma 1 can be made.

Interestingly, **PACP** does not have slice pushouts. Indeed, there is no unique mediation between the squares in the slice of $\underline{1}$:



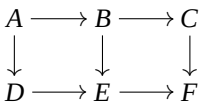
By an idem-relative-pushout [4] we mean the (square) diagram in $\mathbf{C}$ obtained by applying the forgetful functor $U_W : \mathbf{C}/W \to \mathbf{C}$ (which projects $\langle V, r\rangle$ to V) to a pushout diagram in $\mathbf{C}/W$. Let $\mathcal{I}$ denote the class of IPOs in $\mathbf{C}$.

In categories with slice pushouts, it makes sense to talk about IPOs without worrying about particular slices, as the conclusion of the following lemma implies:

Lemma 3. *If* $\mathbf{C}$ *has slice pushouts and a diagram* D *in* $\mathbf{C}/X$ *maps via the forgetful functor* U_X *to an IPO (i.e.,* $U_X D \in \mathcal{I}$*) then* D *is a pushout diagram in* $\mathbf{C}/X$.

Moreover, in categories with slice pushouts, IPOs behave somewhat like ordinary pushouts, as demonstrated by the following lemma.

Lemma 4. *Suppose that* $\mathbf{C}$ *has slice pushouts and the left square is an IPO. Then the entire diagram is an IPO iff the right square is an IPO.*



3.2 Pullbacks in coslices

Dually, a pullback of $f\colon \langle a, A\rangle \to \langle r, W\rangle$ and $g\colon \langle b, B\rangle \to \langle r, W\rangle$ in the coslice category $V/\mathbf{C}$ is, equivalently, a product in $(V/\mathbf{C})/\langle r, W\rangle$. We say that a category has *coslice pullbacks* when it has pullbacks in all of its coslices.

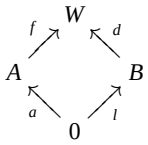


Fig. 1. An IPO corresponding to a label.

Lemma 5. *Free PROPs have coslice pullbacks. **PAP** has coslice pullbacks.*

Proof. Proceed exactly as in Lemmas 1 and 2.

By an idem-relative-pullback (IPB), we mean the (square) diagram obtained from a pullback diagram in a coslice category under the image of the forgetful functor to **C**. We immediately obtain dual versions of Lemmas 3 and 4, the latter of which we state below.

Lemma 6. *Suppose that* **C** *has coslice pullbacks and the right square is an IPB. Then the entire diagram is an IPB iff the left square is an IPB.*

$$\begin{array}{ccccc} A & \longrightarrow & B & \longrightarrow & C \\ \downarrow & & \downarrow & & \downarrow \\ D & \longrightarrow & E & \longrightarrow & F \end{array}$$

3.3 Labels

As we mentioned in the Introduction, IPOs have been used by Leifer and Milner to derive labelled transition systems for calculi equipped with a reduction semantics derived from a set of ground rules. Here we give a brief overview of the technique.

Leifer and Milner's framework of choice is their notion of '*reactive system*,' which consists of a category of contexts with a chosen object 0, a subcategory of evaluation contexts which satisfies certain additional axioms and a set of reduction rules $\mathcal{R}$. The arrows with domain 0 are thought of as closed terms. We shall not give a formal definition here; instead we refer the reader ahead to Definition 3, which deals with a more general situation where reduction rules may be open – to obtain a (closed) reactive system from that definition one needs to assume additionally that the domains of l and r in every rule $\langle l, r\rangle \in \mathcal{R}$ are 0.

Sewell's central idea [16] which guides the definition of the derived LTS is that labels should be certain contexts – more accurately, $a \xrightarrow{f} a'$ when fa (a in the context of f) can perform a single reduction and result in a'. Moreover, as explained in the Introduction, f must be the 'smallest' such context. The notion of IPO gives us a precise way to measure when a context is the smallest. Indeed, consider Fig. 1, where a is an arbitrary term, l is the left hand side of a reduction rule $\langle l, r\rangle \in \mathcal{R}$ and d is an evaluation context. The fact that the diagram is commutative implies that fa can perform a reduction resulting in dr, where the redex l has been replaced by r, the right-hand side of the rule. Requiring the diagram to be an IPO results in an elegant formalisation

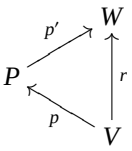
of the fact that f does not contain redundant material, not necessary for the reduction. The LTS determined in this way can be shown to be well-behaved. In particular, if the underlying category has slice pushouts then bisimilarity is a *congruence*, in the sense that $a \sim b$ implies that $ca \sim cb$ for al c in $\mathbf{C}$.

3.4 Category of factorisations

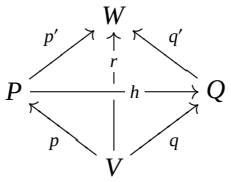
The category of factorisations of an arrow provides a convenient setting for studying slice pushouts and coslice pullbacks which we shall use in the rest of the paper.

Definition 2 (Factorisations). The category $\mathrm{Fact}(\mathbf{C}, r)$ of factorisations of an arrow $r\colon V \to W$ in $\mathbf{C}$ is consists of objects and arrows as defined below.

– objects: commutative diagrams in $\mathbf{C}$ of the form



– arrows: an arrow from $\langle P, p, p'\rangle$ to $\langle Q, q, q'\rangle$ is a commutative diagram in $\mathbf{C}$ of the form



– Composition and identities are obvious.

The following fact is immediate.

Proposition 1. $(V/\mathbf{C})/\langle r, W\rangle \cong \mathrm{Fact}(\mathbf{C}, r) \cong \langle V, r\rangle/(\mathbf{C}/W)$.

Such categories of factorisations form a convenient universe to speak about slice pushouts from $\langle V, r\rangle$ and coslice pullbacks from $\langle r, W\rangle$ in $\mathbf{C}$, since the former are precisely the coproducts and the latter are the products in $\mathrm{Fact}(\mathbf{C}, r)$.

3.5 Generalisation to G-categories

In [11, 13] the second and third author generalised Leifer and Milner's theory to a 2-categorical setting, where structural congruence axioms (usually involving the commutativity of parallel composition) are replaced by invertible 2-cells. The extra structure is necessary, because simply quotienting terms results in structures where IPOs do not exist, as in the case of **PACP** defined in Example 2 (cf. also [11]).

The problem is alleviated by working with G-categories – 2-categories with invertible 2-cells – and considering GIPOs. The latter are the natural bicategorical generalisation of IPOs: namely, rather than pushouts in slice categories, one considers bipushouts in pseudo-slice categories. One can, equivalently, define a category of pseudo-factorisations and consider bicoproducts (obtaining GIPOs) and biproducts (obtaining GIPBs).

Example 3. A *G-PROP* is a PROP with the underlying category carrying the structure of a G-category, i.e., a 2-category with all 2-cells invertible. As an example, consider the PROP **PAP** from Example 2 (see the proof of Lemma 2 for an explicit representation of the arrows of **PAP**). Additionally, a 2-cell from a term t to a term t' is a family, indexed by the nodes of t, of permutations on the sets of their immediate children, such that the application of all these permutations to t yields t'. Note how such 2-cells induce bijections between the sets S_t and S'_t of positions respectively in t and t'. Clearly, all such 2-cells are invertible, hence the theory of GIPOs described in [11, 13] applies. In particular, the lack of slice pushouts in the PROP **PACP** is avoided here: pseudo-slice bipushouts exist in the above G-PROP, which in the following will be denoted **PA2CP**.

4 Hexagons and universality

In this section we set out on a path to extend the technique of LTS derivation to systems where reduction rules are open in the sense that they can be instantiated with arbitrary parameters. In such a setting, we would also like generate labels for possibly open terms. The basic idea is that instead of considering simply the smallest context which allows a reduction, we would like to calculate both a smallest context and the most general parameter at the same time. We discuss a reasonable universal property, the *locally universal hexagon*, or *lux*, referred to by Sewell [15] as hex-RPO. These can be used to generate a labelled transition system with information about both contexts and parameters, reminiscent of work on tile systems [2].

In order to understand this universal property, we consider its relationship with slice pushouts and coslice pullbacks in the underlying category. It is convenient to work in slices of the so-called twisted arrow category. We show in Theorem 1 that a category has luxes if and only if it has slice pushouts, coslice pullbacks and these 'commute.' This result allows us isolate sufficient conditions for luxes to exist. Assuming that the underlying category has mono arrows, we show that bisimilarity is a congruence. Finally, we examine how the theory generalises to G-categories.

Definition 3 (Open reactive system). An open reactive system $\mathbb{C}$ is a triple $\langle \mathbf{C}, \mathbf{D}, \mathcal{R} \rangle$ consisting of:

- a category $\mathbf{C}$ with a distinguished object 0 – we shall usually refer to its arrows as contexts and, specifically, to the arrows with domain 0 as terms;
- a composition reflecting subcategory $\mathbf{D}$ of $\mathbf{C}$ – the arrows of $\mathbf{D}$ are termed *evaluation* contexts;
- a set $\mathcal{R}$ of pairs of arrows of $\mathbf{C}$, so that if $\langle l, r \rangle \in \mathcal{R}$, then the domains and codomains of l and r are equal – we shall refer to $\mathcal{R}$ as the set of reduction rules.

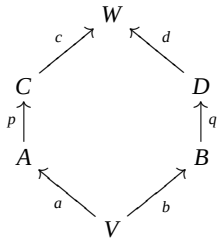


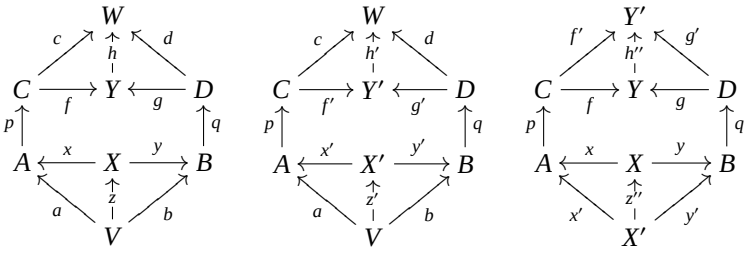
Fig. 2. A hexagon.

Given an open reactive system $\mathbb{C}$, one can define a reduction relation on the terms of $\mathbb{C}$ as follows: $a \longrightarrow\!\!\triangleright\, a'$ if $a = dlx$ and $a' = drx$ for some $d \in \mathbf{D}$, $x \in \mathbf{C}$, and $\langle l, r\rangle \in \mathcal{R}$.

We shall refer to commutative diagrams such as the one illustrated if Figure 2 as commutative hexagons, or simply *hexagons*. The following universal property defines locally universal hexagons, or luxes.

Definition 4 (Luxes). A *locally universal hexagon* (*lux*) for the hexagon of Fig. 2 is a hexagon that factors through it (cf. diagram *i*), and that additionally satisfies a universal property:

> for any other such hexagon (cf. diagram *ii*), there exist unique $h'' : Y \to Y'$ and $z'' : X' \to X$ such that diagram (*iii*) is commutative, $h = h'h''$ and $z = z''z'$.



(*i*) (*ii*) (*iii*)

We denote the lux of diagram (*i*) above as $(p)^{fg}_{xy}(q)$.

We shall say that a category **C** has luxes if every hexagon has a lux. As was the case for IPOs, in categories with luxes one does not need to know which hexagon is a lux for: if a hexagon is a lux, then it is such for all hexagons through which it factors. This property, analogous to Lemma 3, will be proved formally as Lemma 7 below.

In order obtain first intuitions about luxes, let us consider a very simple example. Below we denote string concatenation by ; noted in diagrammatic order (i.e., left-to-right).

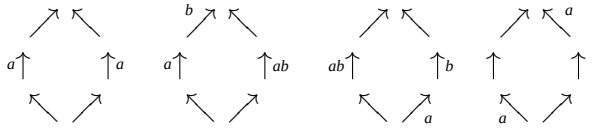


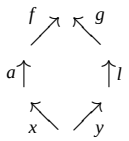
Fig. 3. Luxes in a free monoid.

Example 4. Consider a free monoid over an alphabet Σ viewed as a category **S** with one object. Then luxes exist in **S**. Consider a hexagon as in Fig. 2, where $a; p; c = b; q; d$. Let h be the largest suffix common to c and d, then there exist words f and g so that $c = f; h$ and $d = g; h$. Similarly, let z be the largest prefix common to a and b, then there exist words x and y so that $z; x = a$ and $z; y = b$. Clearly $x; p; f = t; q; g$ and it is straightforward to check that the universal property holds. We now fix $\Sigma = \{a, b\}$ and illustrate several examples of luxes in **S** in Fig. 3.

Armed with the notion of lux, we are ready to define a labelled transition system on possibly open terms.

Definition 5 (LTS). Given an open reactive system $\mathbb{C}$, an LTS can be derived as follows:

- nodes are arbitrary arrows $a : A \to C$ – i.e., domain does not need to be 0: terms are possibly open;
- there is a transition $a \xrightarrow[x]{f} b$ whenever there exist $\langle l, r \rangle \in \mathcal{R}$ and a lux



such that $b = gry$ – thanks to Lemma 7 below, this is well given.

5 Properties of locally universal hexagons

In order to study the properties of luxes, it is convenient to work in *twisted arrow categories*, that we introduce below. Here we give their definition from [6], and examine some of its basic properties. We mention that the category can be concisely described as the category of elements for the homfunctor $\mathbf{C}(-,-) : \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{Set}$.

Definition 6 (Twisted Arrow Categories). Given a category **C**, the twisted arrow category Tw(**C**) has

- objects: the arrows of **C**;

– arrows: an arrow from $f\colon A \to B$ to $f'\colon A' \to B'$ consists of arrows $p\colon B \to B'$ and $q\colon A' \to A$ such that $pfq = f'$; in other words, an arrow from f to f' is a factorisation of f' through f, as in the diagram below.

$$\begin{array}{ccc} B & \xrightarrow{c} & B' \\ {\scriptstyle f}\uparrow & & \uparrow{\scriptstyle f'} \\ A & \xleftarrow[a]{} & A' \end{array}$$

In symbols, we shall use $f \xleftrightarrow[a]{c} f'$ (or $f' \xleftrightarrow[a]{c} f$) to denote such an arrow of Tw(**C**).[3]

As promised, the twisted arrow category gives us a simplified setting in which we may consider the universal property of luxes. Indeed, our first observation is that hexes are in 1-1 correspondence with cospans

$$p \xleftrightarrow[a]{c} r \xleftrightarrow[b]{d} q$$

in Tw(**C**), where $cpa = r = dqb$. Secondly, it is easily verified that luxes are precisely the coproduct diagrams in slices of Tw(**C**).

Proposition 2. *A lux is a hexagon in* **C** *that results from a coproduct diagram in the slice category* Tw(**C**)/r.

Notice that we explicitly talk about the coproduct *diagram* (as opposed to *object*), which includes the cospan formed by the coproduct coprojections.

The following lemma justify us referring to locally universal hexagons (without mentioning which hexagon it is universal with respect to). Thus, when talking about categories with luxes, we shall often abuse notation – in contrast with §3 where we distinguished between slice pushouts and IPOs and coslice pullbacks and IPBs.

Lemma 7. *In a category with luxes, if a hexagon factors through a lux (possibly for another hexagon) then it is a lux for that hexagon.*

Proof. We know that a category has luxes iff every slice Tw(**C**)/r has coproducts. It is straightforward to verify that, given an arbitrary category **C**, when every slice has coproducts, if $\langle A, a\rangle \xrightarrow{f} \langle C, c\rangle \xleftarrow{g} \langle D, d\rangle$ is a coproduct diagram in **C**/X, then for any $X' \in \mathbf{C}$ with $a' : A \to X'$, $b' : B \to X'$ and $c' : C \to X'$ such that $a' = c'f$ and $b' = c'g$ $\langle A, a'\rangle \xrightarrow{f} \langle C, c'\rangle \xleftarrow{g} \langle D, d'\rangle$ is a coproduct diagram in **C**/X'.

In order to obtain further insights into luxes, we shall explore the relationship between slices of Tw(**C**) and the category of factorisations of Definition 2.

First we notice that there is a faithful functor $\mathcal{I}\colon V/\mathbf{C} \to \mathrm{Tw}(\mathbf{C})$ which is the first projection on objects and takes an arrow $h : \langle p, P\rangle \to \langle q, Q\rangle$ to $p \xleftrightarrow[\mathrm{id}]{h} q$. Similarly, there is a functor $\mathcal{J}\colon (\mathbf{C}/W)^{\mathrm{op}} \to \mathrm{Tw}(\mathbf{C})$ which takes $h : \langle P, p'\rangle \to \langle Q, q'\rangle$ to $q' \xleftrightarrow[h]{\mathrm{id}} p'$.

[3] Due to a LaTeX's macro mishap, these are both rendered as $\longleftrightarrow$ in the proceedings version.

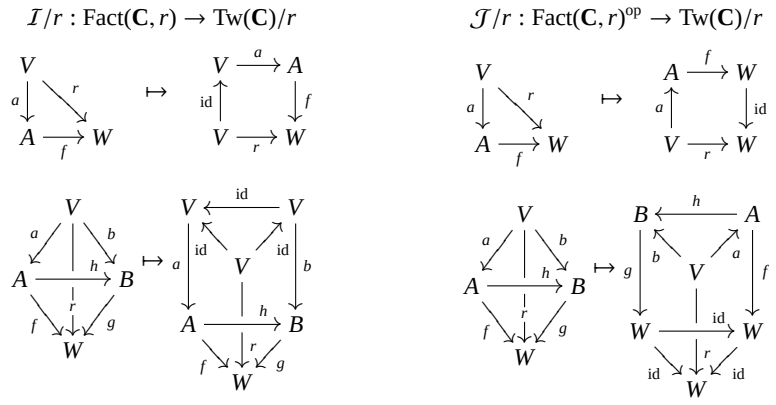


Fig. 4. $\mathcal{I}/r$ and $\mathcal{J}/r$ on objects and arrows.

Both $\mathrm{Fact}(\mathbf{C}, r)$ and $\mathrm{Fact}(\mathbf{C}, r)^{\mathrm{op}}$ can be seen as full subcategories of $\mathrm{Tw}(\mathbf{C})$ via the functors $\mathcal{I}/r\colon\ \mathrm{Fact}(\mathbf{C}, r) \to \mathrm{Tw}(\mathbf{C})/r$ and $\mathcal{J}/r\colon\ \mathrm{Fact}(\mathbf{C}, r)^{\mathrm{op}} \to \mathrm{Tw}(\mathbf{C})/r$. Observe that the second functor is well defined, since $(\mathbf{C}/W)^{\mathrm{op}}/\langle V, r\rangle \cong (\langle V, r\rangle/(\mathbf{C}/W))^{\mathrm{op}}$, for all categories $\mathbf{C}$ and arrows $r\colon V \to W$ in it. We illustrate the actions of $\mathcal{I}/r$ and $\mathcal{J}/r$ in Fig. 4.

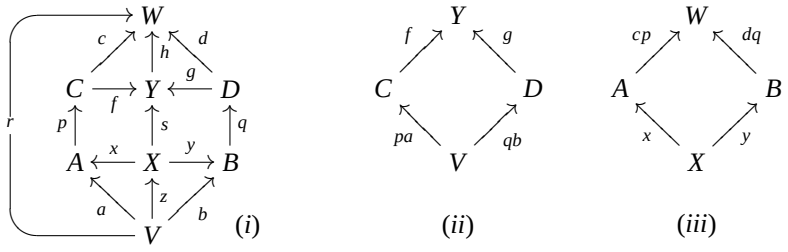
Lemma 8. *$\mathcal{I}/r$ and $\mathcal{J}/r$ have left adjoints, respectively $\Phi\colon\ \mathrm{Tw}(\mathbf{C})/r \to \mathrm{Fact}(\mathbf{C}, r)$ and $\Psi\colon\ \mathrm{Tw}(\mathbf{C})/r \to \mathrm{Fact}(\mathbf{C}, r)^{\mathrm{op}}$.*

It is useful for us to examine the functors Φ and Ψ in more detail. The action of Φ on objects and arrows of $\mathrm{Tw}(\mathbf{C})/r$ is shown in Fig. 5. Note that $\Phi \circ I/r = \mathrm{id}_{\mathrm{Fact}(\mathbf{C},r)}$ and $\Psi \circ J/r = \mathrm{id}_{\mathrm{Fact}(\mathbf{C},r)^{\mathrm{op}}}$. In fact, Lemma 8 states that both $\mathrm{Fact}(\mathbf{C}, r)$ and its opposite are full reflective subcategories of $\mathrm{Tw}(\mathbf{C})/r$.

Corollary 1. *Coproducts in $\mathrm{Tw}(\mathbf{C})/r$ map via Φ to coproducts in $\mathrm{Fact}(\mathbf{C}, r)$, and thus to coproducts in $\langle V, r\rangle/(\mathbf{C}/W)$, which are pushouts in $\mathbf{C}/W$.*

Corollary 2. *Coproducts in $\mathrm{Tw}(\mathbf{C})/r$ map via Ψ to products in $\mathrm{Fact}(\mathbf{C}, r)$, and thus to products in $(V/\mathbf{C})/\langle r, W\rangle$ which are pullbacks in $V/\mathbf{C}$.*

Lemma 9. *A diagram (i) is a coproduct diagram of $p \xleftarrow[a]{c} r$ and $q \xleftarrow[b]{d} r$ in $\mathrm{Tw}(\mathbf{C})/r$ iff (1) diagram (ii) is a pushout in $\mathbf{C}/W$, and (2) diagram (iii) is a pullback in $V/\mathbf{C}$.*



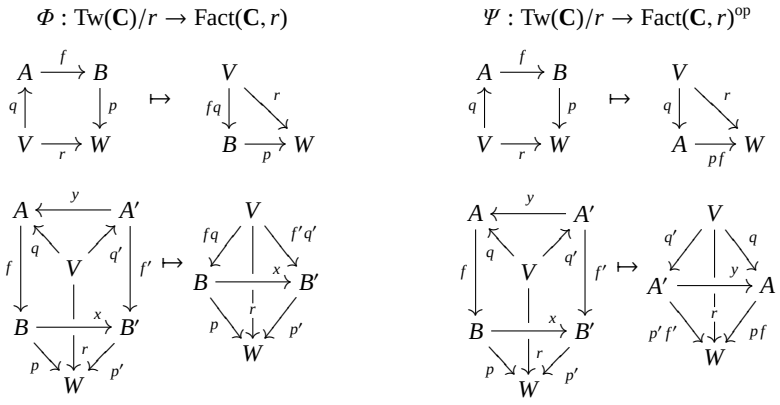
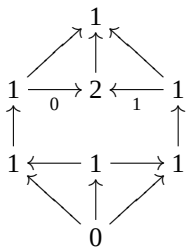


Fig. 5. Φ and Ψ on objects and arrows.

Proof. The only if direction is given by Corollaries 1 and 2. The if direction is easily verified.

Note that Lemma 9 explicitly assumes that the resulting hex is commutative. Consider for instance the following diagram in **Set**:



The lower hexagon results from calculating a local pushout of $0 \to 1$ with itself in $\mathbf{Set}/1 \cong \mathbf{Set}$, while the upper one from a local pullback of $1 \to 1$ with itself in $0/\mathbf{Set} \cong \mathbf{Set}$. Notice that the resulting inner hexagon is *not* commutative.

We shall say that slice pushouts and coslice pullbacks *commute* when, given a commutative square (the outside of Fig. 6), constructing a pushout of a and b in $\mathbf{C}/W$ and a pullback of c and d in $V/\mathbf{C}$ results in an inner commutative diagram ($fx = gy$).

Theorem 1. *A category* **C** *has luxes iff it has slice pushouts, coslice pullbacks and these commute.*

Proof. If **C** has slice pushouts, coslice pullbacks and these commute, then one can explicitly construct a lux, using the conclusions of Lemma 9, since it is easy to show that the commutativity property ensures the commutativity of the resulting hexagon.

Conversely, if **C** has luxes then it is easy to show that it has slice pushouts, i.e., coproducts in Fact($\mathbf{C}, r$) and coslice pullbacks, i.e., products in Fact($\mathbf{C}, r$). Indeed, it is

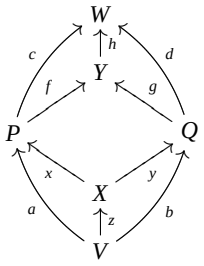
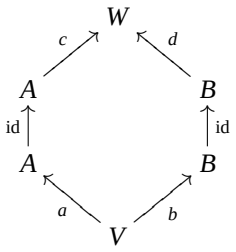


Fig. 6. Commutativity of slice pushouts and coslice pullbacks.

enough to calculate the lux of the hexagon below:

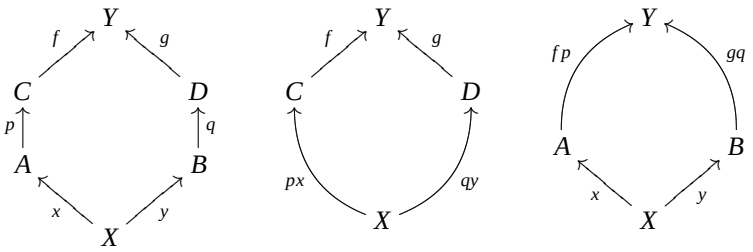


Using the fact that Φ and Ψ preserve coproducts, the resulting lux maps via Φ to the slice pushout of a and b in $\mathbf{C}/W$ and via Ψ to the coslice pullback of c and d in $V/\mathbf{C}$. The commutativity property follows directly.

As an immediate consequence, it follows that **Set** does not have luxes, since the commutativity property is not satisfied.

When working in categories with luxes, we can use the conclusions of Lemma 9 to obtain a characterisation of luxes in terms of IPOs and IPBs.

Lemma 10. *In a category with luxes, a commutative diagram (i) is a lux iff diagram (ii) is an IPO and diagram (iii) is an IPB.*



It is useful to consider properties of **C** that ensure that slice pushouts and coslice pullbacks commute. One obvious such property is that either all arrows of **C** are mono, another is that all arrows of **C** are epi.

Corollary 3. *The following conditions are each sufficient for the existence of luxes in category* **C**.

1. **C** *has slice pushouts, slice pullbacks and all arrows are mono;*
2. **C** *has slice pushouts, slice pullbacks and all arrows are epi.*

Theorem 2. *Free PROPs have luxes.* ***PAP*** *has luxes.*

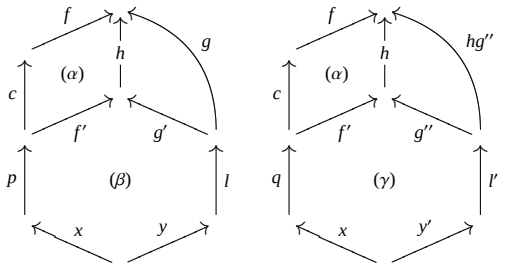
Proof. It is easy shown by induction that all arrows in free PROPs, and all arrows in **PAP**, are mono. This means that no two different terms can be made equal by putting them in the same context. Then use Lemmas 1, 2, and 5.

Theorem 3 (Congruence). *Suppose that $\mathbb{C}$ is an open reactive system. Let $\sim$ denote bisimilarity on the LTS introduced in Definition 5.*

If **C** *has luxes and all arrows of* **C** *are mono, then $\sim$ is a congruence, in the sense that if $p \sim q$, then $cp \sim cq$ for all contexts c in* **C**.

Proof. It is enough to show that $\{\, \langle cp, cq\rangle \mid p \sim q,\ c \in \mathbf{C} \,\}$ is a bisimulation.

Indeed, suppose that $p \sim q$ and $cp \xrightarrow[x]{f} p'$. Then we can find a lux, illustrated as the outside of diagram (i) below, where $\langle l, r\rangle \in \mathcal{R}$ and $p' = gry$.



(*i*) (*ii*)

We now calculate a slice pushout of px and ly in diagram (*i*), resulting in f', g' and h such that $hf' = fc$ and $hg' = g$. Then (β) is an IPB and an IPO in the sense of Lemma 10, using the Lemma yields that (β) is a lux. We obtain $p \xrightarrow[x]{f'} g'ry$

Since $p \sim q$, also $q \xrightarrow[x]{f'} q'$ where $q' \sim g'ry$. Let (γ) be a lux responsible for the transition, so that $\langle l', r'\rangle \in \mathcal{R}$ and $q' = g''r'y'$. Pasting the IPO (α) results in a hexagon which is an IPO. Using the fact that h is mono, it is also an IPB. Thus $cq \xrightarrow[x]{f} hq'$. But $p' = gry = hg'ry$, and since $q' \sim g'ry'$, the proof is complete.

Dually, the following holds.

Proposition 3. *If* **C** *has luxes and all arrows of* **C** *are epi, then $px \sim qx$ for all x in* **C**.

As a consequence of Theorem 2 and the fact that the arrows of free PROPs are mono, the LTS obtained from Definition 5 for any reactive system over a free PROP yields a congruent bisimilarity.

6 Structural congruence as invertible 2-cells

In §3.5 we gave a rough description of how to generalise the concepts of IPOs and IPBs to G-categories. Here we give a brief description of how to generalise the theory of luxes. The definition of G-lux is simple to state.

Definition 7 (G-lux). Given a G-category **C**, the definition of Tw(**C**) can easily be extended to a G-category. The arrows are now twisted squares with a 2-cell, and the 2-cells are 2-cells between the top and bottom components such that everything commutes. A G-lux is a bicoproduct in pseudo-slice category Tw(**C**)/r.

An open G-reactive system is simply an open reactive system on a G-category, the only extra requirement is for the subcategory of evaluation contexts to be full on the 2-dimensional structure.

Given a G-reactive system, it is easy to extend Definition 5 to generate an LTS using G-luxes. One obtains a transition system with possibly open terms as states. It is also possible to consider an LTS where the states are terms quotiented by isomorphism (or, in process calculus terminology, structural congruence) – the congruence theorem holds in both instances; see [17, Ch. 2] for details.

It is fairly straightforward to rework the theory presented in the previous section in this more general setting, but we omit the details here. Using the concepts discussed in §3.5, one obtains generalised versions of Theorem 1, Lemma 10 and Theorem 3. In the latter, the mono requirement is replaced by a 2-categorical version which states that for any arrow f and 2-cells α and β, if $f\alpha = f\beta$ then $\alpha = \beta$.

Proposition 4. ***PA2CP*** *(cf. Example 3) has G-luxes.*

Proof (sketch). The proof follows the general structure of those of Theorem 2 and Lemma 2. To show that **PA2CP** has pseudo-slice bipushouts, consider a 2-cell $\alpha\colon t \Rightarrow t'$ with decompositions $t = ca$, $t' = db$. These decompositions correspond to monotonic functions Λ_{ca}, Λ_{db} as sketched in the proof of Lemma 1. The following function on S_t:

$$\Lambda_1(\rho) = \begin{cases} 1 \text{ if } \Lambda_{ca}(\rho) = 1 \text{ and } \Lambda_{db}(\overline{\alpha}(\rho)) = 1 \\ 2 \text{ if } \Lambda_{ca}(\rho) = 1 \text{ and } \Lambda_{db}(\overline{\alpha}(\rho)) = 2 \\ 3 \text{ if } \Lambda_{ca}(\rho) = 2 \end{cases}$$

(where $\overline{\alpha} : S_t \to S_{t'}$ is the bijection, induced by α, between positions in terms) defines a decomposition $t = xyz$, and moreover $z = a$. Analogously one obtains a decomposition $t' = x'y'z'$, with $z' = b$ and $x' = x$. These decompositions form the 1-cell part of the required pseudo-slice bipushout square; to find the required 2-cells, proceed as in the case of free monoids and permutations in [17].

Example 5. We can construct an open (G-)reactive system on **PA2CP** by letting the set of reduction rules $\mathcal{R}$ be the singleton consisting of the single rule $\langle a.1 \mid \overline{a}.2, 1 \mid 2\rangle$. In the following, we shall use P, Q and X as meta variables which stand for any closed term (arrow $\underline{0} \to \underline{1}$) of **PA2CP**.

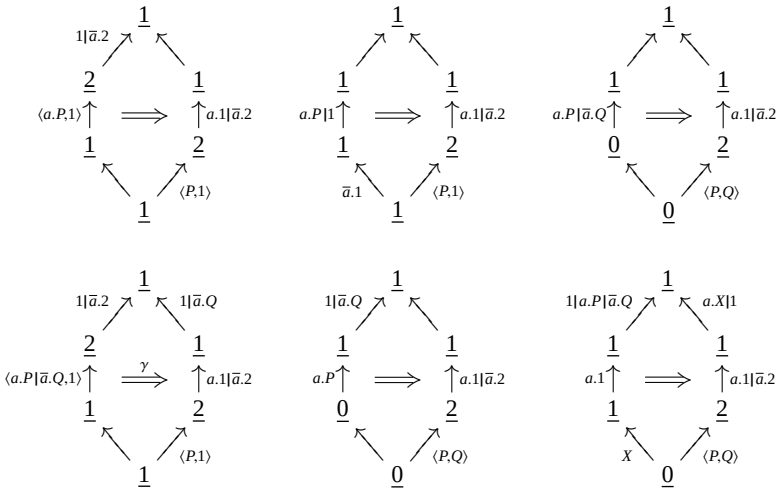


Fig. 7. G-luxes in **PA2CP**.

The subcategory of evaluation contexts is taken to be the smallest composition reflecting 2-full 2-subcategory which includes arrows of the form $\langle 1 \mid P\rangle : \underline{1} \to \underline{1}$. In more intuitive terms, the non-evaluation contexts are precisely the contexts which have a hole under a prefix.

In Fig. 7, we illustrate several examples of G-luxes, which in turn lead to labels of the induced LTS. Thus, the top left diagram gives a transition $\langle a.P, 1\rangle \xrightarrow{1|\bar{a}.2} P \mid 1$, the next diagram leads to a transition $a.P \mid 1 \xrightarrow[\bar{a}.1]{} P \mid 1$. The next transition induced is $a.P \mid \bar{a}.Q \longrightarrow P \mid Q$, which can be seen as internal reduction since no external context or parameter is required. In the second row, the first lux from the left demonstrates the function of the 2-cells in **PA2CP**: here γ is the unique permutation $a.P \mid \bar{a}.Q \mid \bar{a}.1 \to a.P \mid \bar{a}.1 \mid \bar{a}.Q$. The label generated is $\langle a.P \mid \bar{a}.Q, 1\rangle \xrightarrow{1|\bar{a}.2} a.P \mid \bar{a}.1 \mid \bar{a}.Q$.

The final two diagrams illustrate what we believe is the main problem with luxes – indeed, the problem can be observed already in the much simpler Example 4. Roughly, while the universal property of luxes ensures that there is no redundant information in the contexts and in the parameters, there may still be some overlap between contexts and parameters. Indeed, consider the middle diagram of the second row of Fig. 7. The lux leads to the transition $a.P \xrightarrow{1|\bar{a}.Q} P \mid Q$. However, the information in Q is not necessary for this reduction, since it appears both in the context on the left and in the parameter on the right. Unfortunately, since Q is arbitrary, this means that the resulting LTS is infinitely branching. The final diagram is even more redundant, since no part of the term is actually necessary for the reduction; now we have X appearing both as a parameter an the left and as part of the context on the right, and P and Q appearing both as the parameters on the right and as part of the context on the left. This diagram induces the transition $a.1 \xrightarrow[X]{1|a.P|\bar{a}.Q} a.X \mid P \mid Q$.

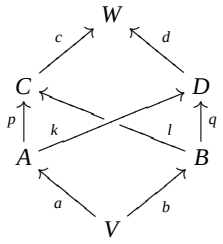


Fig. 8. An irredundant hexagon.

7 Towards a general theory

Let us consider the following simple property, in order to rule out some of the 'redundant' luxes identified in Example 5.

Definition 8 (Irredundant hexagon). A hexagon is said to be irredundant when there exist $k\colon A \to D$ and $l\colon B \to D$ so that all regions of Fig. 8 are commutative; that is $lb = pa$, $cl = dq$, $ka = qb$ and $dk = cp$.

A lux is said *irredundant* when it is irredundant as a hexagon. The property can be extended to cover G-luxes in the obvious way, that is, instead of commutativity one requires the presence of compatible 2-cells.

Example 6. Consider the luxes illustrated in Fig. 3 and discussed in Example 4. It is easy to show that the first three diagrams are irredundant as hexes, but the final one is not. Now consider the G-luxes illustrated in Fig. 7 and discussed in Example 5. Again, all of the luxes apart from the middle and the right lux of the second row are irredundant.

Thus, by considering irredundant luxes, we eliminate the problematic luxes identified in our case studies. The obvious next steps are to alter the LTS definition so that only irredundant luxes are taken into account, and to study bisimilarity on the resulting structures. Alas, the simple-minded modification to Definition 5 will not do: the technique we use to prove our congruence results (viz., Theorem 3 and Proposition 3) does not stand for irredundant luxes alone, as a lux that is a factor of an irredundant lux need not be irredundant itself. Indeed, bisimilarity in general is *not* a congruence. Recalling e.g. the reactive system of Example 5, and consider the possible labels of transitions with domains of the form $a.P$. If a reduction involves $a.P$, then it must occur in context with an output $\overline{a}.Q$, for some Q; this introduces redundancy, as Q must arise as a parameter that instantiates the reduction rule. Thus $a.P$ has *no* irredundant labels, which implies $a.P \sim b.P$, for all $a \neq b$. But $a.P$ can reduce in the presence of an output on a while $b.P$ cannot, and so congruence is broken. As future work, we plan to study ways of deriving transition systems for open terms that at the same time carry no redundancy in the labels, and are sufficient for coinductive reasoning.

In any case, we feel that the framework for deriving labelled transition systems from reductions is still in its infancy, and requires further development. Our ultimate goal is

an abstract method that, when applied to the standard reduction system of a calculus like, say, the π-calculus, yields a labelled transition system on which bisimilarity is a congruence, and moreover: (1) gives rise to feasible coinductive techniques, and (2) is fully abstract with respect to standard equivalences defined in terms of contextual closures (such as barbed congruence). The final theory will necessarily involve a satisfactory treatment of variables, parameters and parametric rules. We believe that luxes and irredundancy, introduced in this paper, shall serve as important tools in our future research on 'labels from reductions.'

References

1. H. Ehrig and B. König. Deriving bisimulation congruences in the DPO approach to graph re writing. In *Proceedings of FoSSaCS'04, 7th International Conference on Foundations of Software Science and Computation Structures*, volume 2987 of *LNCS*, pages 151–166, 2004.
2. F. Gadducci and U. Montanari. The tile model. In G. Plotkin, C. Stirling, and M. Tofte, editors, *Proof, Language and Interaction: Essays in Honour of Robin Milner*, pages 133–166. MIT Press, 2000.
3. O. H. Jensen and R. Milner. Bigraphs and mobile processes. Technical Report 570, University of Cambridge, 2003.
4. J. Leifer and R. Milner. Deriving bisimulation congruences for reactive systems. In *International Conference on Concurrency Theory, CONCUR'00*, volume 1877 of *LNCS*, pages 243–258, 2000.
5. S. Mac Lane. Categorical algebra. *Bulletin of the American Mathematical Society*, 71:40–106, 1965.
6. S. Mac Lane. *Categories for the Working Mathematician*. Graduate Texts in Mathematics. Springer Verlag, 2nd edition, 1992.
7. R. Milner. Calculi for interaction. *Acta Informatica*, 33(8):707–737, 1996.
8. R. Milner. Bigraphical reactive systems. In *International Conference on Concurrency Theory, CONCUR'01*, volume 2154 of *LNCS*, pages 16–35, 2001.
9. R. Milner. Bigraphs for Petri nets. In *Lectures on Concurrency and Petri Nets 2003*, volume 3098 of *Lecture Notes in Computer Science*, pages 686–701, 2004.
10. G. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Aarhus University, Computer Science Department, 1981.
11. V. Sassone and P. Sobociński. Deriving bisimulation congruences using 2-categories. *Nordic Journal of Computing*, 10(2):163–183, 2003.
12. V. Sassone and P. Sobociński. A congruence for Petri nets. In *Workshop on Petri Nets and Graph Transformation*, volume 127 of *ENTCS*, pages 107–120, 2005.
13. V. Sassone and P. Sobociński. Locating reaction with 2-categories. *Theoretical Computer Science*, 333(1-2):297–327, 2005.
14. V. Sassone and P. Sobociński. Reactive systems over cospans. *Proceedings of Logics in Computer Science, LICS 2005*, IEEE Press 2005.
15. P. Sewell. Working notes PS15–PS19, 2000. Unpublished notes.
16. P. Sewell. From rewrite rules to bisimulation congruences. *Theoretical Computer Science*, 274(1–2):183–230, 2002.
17. P. Sobociński. *Deriving process congruences from reaction rules*. PhD thesis, BRICS, University of Aarhus, 2004.