Sortings for Reactive Systems*

Lars Birkedal, Søren Debois, and Thomas Hildebrandt

IT University of Copenhagen
{birkedal,debois,hilde}@itu.dk

Abstract. We investigate *sorting* or *typing* for Leifer and Milner's reactive systems. We focus on transferring congruence properties for bisimulations from unsorted to sorted systems. Technically, we give a general definition of sorting; we adapt Jensen's work on the transfer of congruence properties to this general definition; we construct a *predicate sorting*, which for any decomposible predicate P filters out agents not satisfying P; we prove that the predicate sorting preserves congruence properties and that it suitably retains dynamics; and finally, we show how the predicate sortings can be used to achieve *context-aware reaction*.

1 Introduction

The last decade has seen a series of definitions of reactive systems for which it is possible to derive labeled transition systems with an associated bisimulation relation that is guaranteed to be a congruence relation [1–8]. Sewell proposed to use suitable contexts of the reactive system as labels in the derived labeled transition system [1]. Leifer and Milner refined this approach by suggesting that it suffices to consider minimal contexts, with minimality captured by the notion of *relative pushout* (RPO) in the category corresponding to the reactive system [2]. Milner and Jensen suggested further refinements in their work on bigraphical reactive systems, technically by representing the reactive systems as quotients of precategories, which in turn possess the requisite relative pushouts [3–5]. An alternative approach using 2-categories was suggested by Sassone and Sobocinski [6, 7], and subsequently transferred to double categories by Bruni, Gadducci, Montanari and Sobocinski in [8].

One aim of these abstract definitions of reactive systems is to unify and generalize existing calculi for concurrency and mobility, by providing a uniform behavioral theory: the congruential bisimulation relation associated with the derived labeled transition system. For bigraphical reactive systems, this aim has been evaluated with encouraging results: existing behavioral theories have been recovered for CCS [5], π-calculus [9], and mobile ambients [9]; and bigraphical semantics has contributed to that of Petri-nets [10] and Homer [11].

Bigraphical reactive systems aim also to model aspects of ubiquitous systems directly. An evaluation of this aim was initiated in [12].

* This work was funded in part by the Danish Research Agency (grant no.: 2059-03-0031) and the IT University of Copenhagen (the LaCoMoCo project).

A *sorting* for a reactive system is analogous to a typing discipline for terms: Each sort gives an abstract view of its morphisms, in the same way that each type gives an abstract view of its terms. Various notions of sorting have turned out to be useful for both the meta-modeling aim and for the ubiquitous system aim.

1. In representations of existing calculi in bigraphical reactive systems, sortings remove "junk" morphisms — morphisms not representing anything. These are removed to get a tight correspondence between the bisimulation derived in bigraphs and the intended bisimulation [10, 5, 9, 11].
2. For the modeling of context-aware systems, sortings help restricting reaction rules to apply only in certain contexts, to get "context-aware reaction rules" [13, 12].

The sortings used in *loc.cit.* are all defined by first adding sorts to each object in the category of bigraphs, second stipulating a well-sortedness condition using this extra information, and finally declaring that we will only consider well-sorted morphisms. (Notice again the analogy to typing disciplines.) For representation applications (Item 1 above), sorts and conditions are chosen to make well-sorted all but the junk morphisms. For modeling applications (Item 2 above), sorts are used simply to distinguish sets of contexts; by choosing an appropriate sort for a reaction, we restrict it to specific contexts.

However, we cannot tinker arbitrarily with our underlying category; we must preserve relative pushouts in order to keep bisimulation a congruence. In each example cited above, this preservation property is shown by hand. Moreover, sorting is itself defined explicitly in each case: both Jensen [9] and Milner [5] define sorting for bigraphical place graphs; and Leifer and Milner define bigraphical link graph sorting in [10].

In this paper we investigate sortings for reactive systems and make the following contributions.

1. We give a general definition of sorting, encompassing all the different notions seen in the above examples (Definition 4).
2. We lift Jensen's safety theorem to this general setting (Theorem 12). Jensen's safety theorem gives a sufficient condition under which RPOs may be transferred between sorted and unsorted worlds, but only in the setting of bigraphical place-graph sorting [9].
3. We present a general construction of sorting, the *predicate sorting* (Definition 15). For any predicate P which is preserved under under de-composition, this sorting filters out morphisms not satisfying P.
4. We prove that predicate sortings transfer RPOs (Theorem 20). Thus, if the bisimulation of an unsorted system is a congruence, then so is the bisimulation of the corresponding predicate-sorted system.
5. We prove a *correspondence theorem* (Theorem 25) for predicate sortings: A predicate sorted system suitably preserves the dynamics of its unsorted counterpart.

6. We show that predicate sortings can be used to model some context-aware reaction systems, notably those where some reaction rules should apply only in contexts which do not contain a given sub-context (Theorem 30).

Our setting is reactive systems over categories rather than precategories (the home of bigraphs) or 2-categories. We believe the extension of our work to either setting to be straightforward, but have yet to justify that belief.

This paper is an abridged version of the technical report [14]; refer to that report for omitted proofs.

Overview. In Section 2, we recall Leifer and Milner's reactive systems; in Section 3, we give our general definition of sorting and lift Jensen's transfer theorem; in Section 4, we define predicate sortings; in Section 5, we prove that predicate sortings transfers RPOs; in Section 6, we prove the correspondence theorem; in Section 7, we demonstrate that predicate sortings can be used to define context-aware reaction rules; and in Section 8, we conclude.

Notation and terminology. We will need a tiny bit of standard terminology from the study of (op-)fibrations (see, e.g., [15]). Let $p : \mathbb{E} \to \mathbb{B}$ be a functor. A morphism of $\mathbb{B}$ has a *lift at* E iff it is the p-image of a morphism $f : E \to X$. A morphism f is *above* $p(f)$. A morphism ϕ is *vertical* if it is above an identity. The verticals above a particular identity id_B forms a category, the *fibre* over B. A morphism f is *opcartesian* iff whenever h, f is a span and h is above $g \circ p(f)$, then there exists a unique $\bar{g}$ s.t. $p(\bar{g}) = g$ and $h = \bar{g} \circ f$. (Two morphisms f, g form a *span* if they share domain, a *cospan* if they share codomain.)

2 Reactive Systems

We give a brief introduction to Leifer and Milner's reactive systems [2]. First, terminology and a little intuition. Let $\mathbb{B}$ be a category, and let ϵ be a distinguished object of $\mathbb{B}$. We shall think of morphisms with domain ϵ as *agents* and all other morphisms as *contexts*. Notice that the composition $C \circ a$ of a context $C : X \to Y$ with an agent $a : \epsilon \to X$ yields an agent $C \circ a : \epsilon \to Y$. A *reaction rule* (l, r) is a span of agents, i.e., both $l : \epsilon \to X$ and $r : \epsilon \to X$ for some X. Intuitively, l and r are the left- and right-hand sides of rewrite rule. A set $\mathcal{R}$ of reaction rules gives rise to a *reaction relation*, $\longmapsto$, by closing reaction rules under contexts:

$$a \longmapsto b \quad \text{iff} \quad \exists C \in \mathbb{B}, \exists (l, r) \in \mathcal{R}.\ a = C \circ l, b = C \circ r. \tag{1}$$

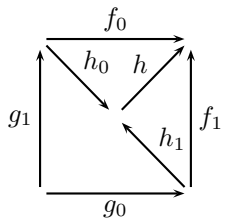
Altogether, these components constitute a reactive system.

Definition 1 (Reactive systems). *A reactive system over a category $\mathbb{B}$ comprises a distinguished object ϵ and a set $\mathcal{R}$ of reaction rules; the reaction rules gives rise to a reaction relation by (1) above.*

Thus far, we have merely restated well-known concepts in the language of category theory. The contribution of Leifer and Milner is their method for deriving labeled transitions from *any* reactive system: Provided the underlying

category has sufficient structure, the bisimulation on these labeled transitions is guaranteed to be a congruence. To give the labeled transitions, we will need the concept of relative pushouts (RPOs).

Definition 2 (Relative pushout). *Consider the following diagram.*



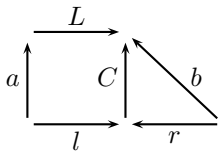
Suppose the outer square commutes. The triple (h_0, h_1, h) *is an RPO for* g_0, g_1 *to* f_0, f_1 *iff the entire diagram commutes and* (h_0, h_1, h) *is universal, that is, if* (h'_0, h'_1, h') *has* $h'_0 \circ g_0 = h'_1 \circ g_1$ *and* $f_i = h' \circ h'_i$*, then there exists a unique* k *s.t.* $h = h' \circ k$ *and* $h_i = k \circ h'_i$*. If* (f_0, f_1, id) *is an RPO for* g_0, g_1 *to* f_0, f_1*, we say that* (f_0, f_1) *is an* idem pushout *(IPO) for* g_0, g_1*.*

(For category-theory buffs: The RPO for g_i to f_i is a pushout of appropriate g_i in the slice-category over the codomain of the f_i.)

Intuitively, if (h_i, h) is an RPO for g_i to f_i, then h is the common part of the contexts f_i. The universality condition says that h is as big as possible: If h' is an alternative common part, then it must factor h, and there are thus commonalities in the f_i captured by h but not by h'. With this intuition, if f_i is an IPO for g_i, the f_i are minimal contexts making up for the differences between the g_i.

Leifer and Milner proceed to construct their labeled transition systems by taking as labels such minimal contexts enabling reaction.

Definition 3. *For a reactive system* $(\mathcal{R}, \epsilon)$ *over* $\mathbb{B}$*, we define the* standard transition relation $\longrightarrow$ *by taking a* $\xrightarrow{L}$ b *iff there exists a context* C *and a reaction rule* $(l, r) \in \mathcal{R}$ *s.t. the following diagram commutes, and the square is an IPO.*



As mentioned, if $\mathbb{B}$ has all RPOs, then the bisimulation induced by the standard transitions is a congruence [2].

3 Sortings

The process of adding sort information, then removing morphisms based on that information is really the construction of a category $\mathbb{E}$, based on some existing category $\mathbb{B}$. There is obviously a forgetful functor $p : \mathbb{E} \to \mathbb{B}$ which is surjective on objects; both Jensen [9] and Milner/Leifer [10] note so. Clearly, this functor characterizes the sorting — Milner and Leifer states: "We shall often confuse [a sorting] with its functor" [10, p.44]. Hence, we suggest taking the existence of such a functor as the *definition* of a sorting.

Definition 4 (Sorting). *A* sorting *of a category $\mathbb{B}$ is a functor into $\mathbb{B}$ that is faithful and surjective on objects.*

It is perhaps helpful to think of a sorting functor p as a refinement of homsets: A homset $\mathbb{B}(B, B')$ is refined into the homsets $\mathbb{E}(E, E')$, where each E and E' are p-preimages of B and B'. Because p is surjective on objects, every homset of $\mathbb{B}$ is so refined; because p is faithful, each such refined homset simply consists of a subset of the morphisms of the original homset.

We are interested in sortings that allow us to infer the existence of RPOs in $\mathbb{E}$ from the existence of RPOs in $\mathbb{B}$. Jensen gives a sufficient condition, *safety*, for making such inferences. However, Jensen formulates safety in the setting of bigraphical place-graph sortings, so we would like to lift Jensen's definition of safety and his RPO-transfer theorem [9, Theorem 4.32] to our general definition of sorting. Remarkably, *virtually nothing needs to be done*: Jensen's definition, theorems and proofs are all formulated exclusively in terms of the (induced) forgetful functor p, so we may transfer his work verbatim to our more general setting. Thus, Definition 5 and Theorems 6, 8, and 12 are essentially due to Jensen, although our formulations are much more general than his[1]. Jensen's proofs of Theorems 6, 8, and 12 can be found either in Jensen's forthcoming thesis [9] or (restated more verbosely) in [14].

Definition 5 (Transfer of RPOs). *A functor $p : \mathbb{E} \to \mathbb{B}$ transfers RPOs iff whenever the p-image of an $\mathbb{E}$-square s has an RPO, then that RPO has a p-preimage that is an RPO for s.*

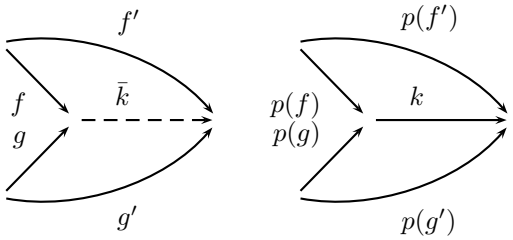
This definition is sufficient to infer the existence of RPOs in $\mathbb{E}$ from the existence of RPO in $\mathbb{B}$:

Theorem 6. *If $\mathbb{B}$ has RPOs and $p : \mathbb{E} \to \mathbb{B}$ transfers RPOs, then $\mathbb{E}$ has RPOs and p preserves RPOs.*

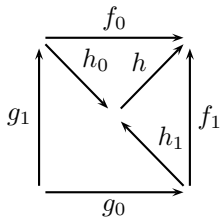
In order to characterize RPOs in $\mathbb{E}$, we have concocted the following generalization of "opcartesian". The notion is inspired by Jensen's notion of minimally sorted sets of morphisms; it is a vehicle for transferring factorization of contexts from $\mathbb{B}$ to $\mathbb{E}$.

[1] In the words of Poincaré [16, p. 34]: "When language has been well-chosen, one is astonished to find that all demonstrations made for a known object apply immediately to many new objects: nothing requires to be changed, not even the terms, since the names have become the same."

Definition 7 (Jointly opcartesian). *Let $p : \mathbb{E} \to \mathbb{B}$ be a functor. A cospan f, g in $\mathbb{E}$ is said to be* jointly opcartesian *iff whenever f', g' is a cospan, f, f' is a span, and g, g' is a span (see the diagram below, left side) with $p(f') = k \circ p(f)$ and $p(g') = k \circ p(g)$ (see the diagram below, right side), then there exists a unique lift $\bar{k}$ of k s.t. $f' = \bar{k} \circ f$ and $g' = \bar{k} \circ f'$.*



Theorem 8. *If $\mathbb{B}$ has RPOs and $p : \mathbb{E} \to \mathbb{B}$ transfers RPOs, then the diagram below is an RPO in $\mathbb{E}$ iff its p-image is an RPO and h_0, h_1 are jointly opcartesian.*



Intuitively, an RPO is the best way to factor a square; h_0, h_1 jointly opcartesian ensures that this best factorization can be lifted from $\mathbb{B}$ to $\mathbb{E}$.

Now that we have a characterization of IPOs for sortings that transfer RPOs sortings, we look for a way to establish that a sorting actually does transfer RPOs. The following generalization of the notion of opfibration will do.

Definition 9 (Weak opfibration). *A functor $p : \mathbb{E} \to \mathbb{B}$ is a* weak opfibration *iff whenever a morphism f of $\mathbb{B}$ has a lift at E, it has an opcartesian lift at E.*

This definition relaxes the requirement of an opfibration (see, e.g., [15]), where each morphism of $\mathbb{B}$ must have an opcartesian lift at each preimage of its domain. However, it does retain the key property that every morphism can be written as the composition of a vertical and an opcartesian.

Proposition 10. *Suppose $p : \mathbb{E} \to \mathbb{B}$ is a sorting. Then p is a weak opfibration iff every morphism f of $\mathbb{E}$ can be written $f = \phi \circ f'$ where ϕ is a vertical and f' is opcartesian.*

Proof. "$\Longrightarrow$". For any $f : E \to E' \in \mathbb{E}$, $p(f)$ must have an opcartesian lift at E, say $\bar{f}$. But $p(\bar{f}) = p(f)$ factors $p(f) \circ \mathrm{id}$, so for some vertical ϕ, $f = \phi \circ \bar{f}$. "$\Longleftarrow$". Any lift $f : E \to E'$ of $p(f)$ can be written $f = \phi \circ \bar{f}$, ϕ vertical and $\bar{f}$ opcartesian, whence $\bar{f}$ is the requisite opcartesian lift. □

Definition 11 (Reflects prefixes, Vertical pushouts). *A functor* $p : \mathbb{E} \to \mathbb{B}$ reflects prefixes *iff whenever* f *is above* $g \circ h$ *then* h *has a lift at the domain of* f*;* p *has* vertical pushouts *iff the fibres have pushouts and such pushouts are also pushouts in* $\mathbb{E}$*.*

Theorem 12. *Let* $p : \mathbb{E} \to \mathbb{B}$ *be a sorting. If* p *is a weak opfibration, reflects prefixes, and has vertical pushouts, then* p *transfers RPOs. If* $\mathbb{B}$ *also has RPOs, then* $\mathbb{E}$ *has RPOs and* p *preserves RPOs.*

We note that sortings can be composed by composing their functors, and we can form conjunctions of sortings by taking their pullbacks. Composition preserves RPO-transfer, and pullbacks preserve both RPO-transfer and the preconditions for Theorem 12; refer to [14] for proofs and details.

4 Predicate Sortings

The example sortings referenced in the introduction are all intended to ban morphisms from the underlying category $\mathbb{B}$. The adding of sort information is but a means to this end; in each case, the authors construct a category $\mathbb{E}$ which resembles $\mathbb{B}$, except that morphisms not satisfying some predicate P are no longer present. We have identified a common feature of these sortings: When read as predicates on morphisms of $\mathbb{B}$, they all define *de*-composible predicates.

Definition 13. *A predicate* P *on the morphisms of a category* $\mathbb{B}$ *is* decomposible *iff* $P(f \circ g)$ *implies* $P(f)$ *and* $P(g)$*.*

This commonality may appear remarkable, but it is not, really, once we realize that the decomposible predicates are precisely those that disallow morphisms that are factored by morphisms in some given set.

Proposition 14. *A predicate* P *on the morphisms of a category* $\mathbb{B}$ *is decomposible iff there exists a set* Φ *of* $\mathbb{B}$*-morphisms s.t.* $P(f)$ *iff for any* g, ψ, h*,* $f = g \circ \psi \circ h$ *implies* $\psi \notin \Phi$*.*

Proof. Suppose P decomposible; take $\Phi = \{\phi \mid \neg P(\phi)\}$. If $P(f)$ and $f = g \circ \psi \circ h$, then $P(\psi)$, so $\psi \notin \Phi$. If $f = g \circ \psi \circ h$ implies $\psi \notin \Phi$, then $f = \mathrm{id} \circ f \circ \mathrm{id}$, so $f \notin \Phi$, thus $P(f)$. Define instead for some Φ, $P(f)$ iff $f = g \circ \psi \circ h$ implies $\psi \notin \Phi$. If $P(f \circ g)$ and, say, $f = h \circ \phi \circ i$, then $f \circ g = h \circ \phi \circ (i \circ g)$, so $\phi \notin \Phi$. □

In an encoding $[\![-]\!]$ of a calculus as a reactive system, it is natural to take Φ to be the complement of the image of the encoding $[\![-]\!]$. However, the resulting predicate is different from just defining "$P(f)$ iff f is in the image of $[\![-]\!]$"; the former definition always allows decompositions of morphisms in the image of $[\![-]\!]$ where as the latter does so only if $[\![-]\!]$ is closed under decomposition in the first place. Because the encodings listed in the introduction all use sortings that are manifestations of decomposible predicates, it appears that so far, images of encodings either turn out to closed under decomposition, or can be closed under decomposition without adversely affecting the resulting bisimulation.

Proposition 14 gives a connection to BiLog [17, 18], a spatial logic for bigraphs. Given a BiLog formula ψ which characterizes a set Ψ of unwanted morphisms, the BiLog formula $(\neg\psi)^{\forall\circ}$ characterizes the morphisms f s.t. $f = x \circ \phi \circ y$ implies $\phi \notin \Psi$. By Proposition 14, the set of morphisms satisfying $(\neg\psi)^{\forall\circ}$ is decomposible, and thus gives rise to a *predicate sorting* as defined below.

We proceed to construct, for any decomposible predicate P on a category $\mathbb{B}$, a corresponding sorting $p : \mathbb{E} \to \mathbb{B}$. The problem we face when constructing any sorting is that we would like to retain as many morphisms of $\mathbb{B}$ as possible, while guaranteeing that we never inadvertently violate P by composition. Suppose for instance that we have morphisms $f : A \to B$ and $g : B \to C$, and that we have both $P(f)$ and $P(g)$, but *not* $P(g \circ f)$. We must either disallow f and allow g, or allow f and disallow g. In the predicate sorting, we retain both options: As preimage of an object B, we take all pairs (X, Y) of sets of morphisms into and out of B such that every morphism in X can safely be composed with every morphism in Y.

Definition 15 (Predicate sorting). *Let $\mathbb{B}$ be a category, and let P be a decomposible predicate on the morphisms of $\mathbb{B}$; we define the* predicate sorting *$p : \mathbb{E} \to \mathbb{B}$ for P. The category $\mathbb{E}$ has pairs (X, Y) as objects, where, for some object B of $\mathbb{B}$, X is a set of $\mathbb{B}$ morphisms with codomain B and Y is a set of $\mathbb{B}$-morphisms with domain B, subject to the following conditions.*

$$\mathrm{id}_B \in X, Y \qquad \text{(ID)}$$
$$f \in X \cup Y \implies P(f) \qquad \text{(SOUND)}$$
$$f \in X, g \in Y \implies P(g \circ f) \qquad \text{(COMP)}$$
$$g \circ f \in X \implies g \in X \qquad \text{(SUFFIX)}$$
$$g \circ f \in Y \implies f \in Y \qquad \text{(PREFIX)}$$

There is a morphism $f : (X, Y) \to (U, V)$ whenever the following holds.

$$f \in Y, f \in U \qquad \text{(VALID)}$$
$$x \in X \implies f \circ x \in U \qquad \text{(PRESERVE)}$$
$$v \in V \implies v \circ f \in Y \qquad \text{(REFLECT)}$$

We put this definition in words. For an object (X, Y), we require that X, Y contain the identity (ID); that morphisms in X, Y satisfy P (SOUND); that morphisms in X, Y are composible (COMP); that X is suffix-closed (SUFFIX); and that Y is prefix-closed (PREFIX). The first three requirements picks out all possible combinations of morphisms satisfying P. The latter two requirements ensures the existence of opcartesians and that the sorting reflects prefixes, respectively; we will need these properties to transfer RPOs. Notice that decomposibility of P is integral only to these latter two requirements.

For a morphism f, we require that it is contained in the sets at its domain and codomain (VALID); that it preserves validity of its domain (PRESERVE); and that it reflects validity at its codomain (REFLECT). The latter two requirements ensure

that we do not accidentally violate P by successive compositions. (Technically, we could do without (VALID), which follows from (PRESERVE), (REFLECT) and (ID); we feel that the definition is clearer as it stands.)

5 Transfer Theorem for Predicate Sortings

In this section, we prove that a predicate sorting $p : \mathbb{E} \to \mathbb{B}$ transfers RPOs. First, we establish that each fibre is a lattice (Proposition 16); second, we characterize the opcartesians (Definition 17 and Proposition 18); third, we use this characterization to show that p is a weak opfibration (Proposition 19); and fourth, we show that p transfers RPOs (Theorem 20). First, each fibre is a lattice.

Proposition 16. *If $\phi : (X, Y) \to (U, V)$ is vertical, then $X \subseteq U$ and $Y \supseteq V$. Ordered pointwise under $\subseteq$ and $\supseteq$, each fibre is a lattice with joins $(X, Y) \sqcup (U, V) = (X \cup U, Y \cap V)$ and meets $(X, Y) \sqcap (U, V) = (X \cap U, Y \cup V)$.*

We characterize the opcartesians. For a morphism $f : A \to B$ and a preimage (X, Y) of A, we use (PRESERVE) and (REFLECT) to *define* a preimage of B.

Definition 17. *Let $f : A \to B$ be a morphism of $\mathbb{B}$, and let $X \subseteq \{g \mid \mathrm{cod}(g) = A\}$ and $Y \subseteq \{h \mid \mathrm{dom}(h) = A\}$. We define operators $\bullet$ and $\circ$ by $f \circ X = \{f \circ x \mid x \in X\}$ and $Y \bullet f = \{g \mid g \circ f \in Y\}$. For any set Z of morphisms, we define the* suffix *and* prefix closures *$Z^{\mathbf{s}} = \{h \mid \exists g.\ h \circ g \in Z\}$ and $Z^{\mathbf{p}} = \{g \mid \exists h.\ h \circ g \in Z\}$.*

Proposition 18. *A morphism $f : (X, Y) \to (U, V)$ is opcartesian if and only if $U = (f \circ X)^{\mathbf{s}}$ and $V = (Y \bullet f)^{\mathbf{p}}$.*

Proposition 19. *A predicate sorting $p : \mathbb{E} \to \mathbb{B}$ is a weak opfibration.*

It is straightforward to establish that p reflects prefixes and has vertical pushouts; see [14]. Thus, by Theorem 12, we have the desired transfer theorem.

Theorem 20. *If $\mathbb{B}$ has RPOs, then a predicate sorting $p : \mathbb{E} \to \mathbb{B}$ transfers RPOs.*

6 Correspondence Theorem for Predicate Sortings

Taking the view that sortings exist to get rid of junk morphisms, when is a sorting good enough? Not just any sorting will do. For instance, for any category $\mathbb{B}$ and predicate P, we can construct a category $\mathbb{E}$ that has, for each f with $P(f)$, unique objects f_X, f_Y and a morphism $f : f_X \to f_Y$. This category gives a sorting $p : \mathbb{E} \to \mathbb{B}$ that transfers RPOs and has as image precisely the morphisms f with $P(f)$, but surely, this sorting is untenable: It supports no non-trivial compositions, reactions, or transitions. We believe that a sorting will prove usable if our chosen reactive system in $\mathbb{B}$ and restricted to morphisms satisfying P can be recovered in $\mathbb{E}$, and similarly for transitions. We establish that our predicate

sortings maintain this correspondence between reactions and transitions in Theorem 25 below. First, we must make our notion of correspondence precise. For a predicate sorting $p : \mathbb{E} \to \mathbb{B}$, we let $\mathbb{E}$ *inherit* reactions from $\mathbb{B}$. Inheritance will in turn require a lift of the distinguished object ϵ, the domain of agents.

Lemma 21. *Write $\bar{\epsilon}$ for the pair $((\mathrm{id}_\epsilon)^{\mathbf{s}}, \{f : \epsilon \to X \mid P(f)\})$. Then $\bar{\epsilon}$ is an object above ϵ, and any morphism $f : \epsilon \to X$ with $P(f)$ has a lift at $\bar{\epsilon}$.*

Definition 22 (p-inherited reactive system). *Let $p : \mathbb{E} \to \mathbb{B}$ be a sorting, and let $\mathcal{R}$ be a ground reactive system on $\mathbb{B}$. The* p-inherited reactive system *has distinguished object $\bar{\epsilon}$ and reaction rules $\bar{\mathcal{R}}$ defined by*

$$\bar{\mathcal{R}} = \{(f, g) \mid f, g : \bar{\epsilon} \to X \text{ for some } X, \text{ and } (p(f), p(g)) \in \mathcal{R}\}.$$

Conversely, reactions and transitions in $\mathbb{E}$ can be translated to $\mathbb{B}$.

Definition 23 (p-induced reactions and transitions). *Let $p : \mathbb{E} \to \mathbb{B}$ be a sorting, and let $\longmapsto$ be a reaction relation on $\mathbb{E}$. We define the* p-induced reaction relation $[\![\longmapsto]\!]$ *in $\mathbb{B}$ by taking for any f, g,*

$$p(f)\ [\![\longmapsto]\!]\ p(g) \quad \textit{iff} \quad f \longmapsto g.$$

Let $\longrightarrow$ be the corresponding transition relation. We define the p-induced transition relation $[\![\longrightarrow]\!]$ *in $\mathbb{B}$ by taking for any f, g, h,*

$$p(f)\ [\![\xrightarrow{p(h)}]\!]\ p(g) \quad \textit{iff} \quad f \xrightarrow{h} g.$$

Having moved reactions up, and reactions and transitions back down, we compare the result to restricting the original $\mathbb{B}$ reactions to P.

Definition 24 (P-restricted reactions and transitions). *Let $\longmapsto$ be a reaction relation. We define the* P-restricted reaction relation $\lfloor\longmapsto\rfloor$ *(in the obvious way) by*

$$f\ \lfloor\longmapsto\rfloor\ g \quad \textit{iff} \quad f \longmapsto g \text{ and } P(f), P(g).$$

Let $\longrightarrow$ be the corresponding transition relation. We define the P-restricted transition relation $\lfloor\longrightarrow\rfloor$ *by*

$$f\ \lfloor\xrightarrow{h}\rfloor\ g \quad \textit{iff} \quad f \xrightarrow{h} g \text{ and } P(f), P(g), P(h), P(h \circ f).$$

Theorem 25 (Correspondence). *Let $\mathbb{B}$ be a category with RPOs, let P be a decomposible predicate, let $p : \mathbb{E} \to \mathbb{B}$ be the predicate sorting for P, let $\mathcal{R}$ be a reactive system on $\mathbb{B}$, and let $\bar{\mathcal{R}}$ be the p-inherited reactive system on $\mathbb{E}$. Then*

1. *the p-induced and P-restricted reaction relations coincide, and*
2. *the p-induced and P-restricted transition relations coincide.*

To prove the correspondence theorem, we will need better understanding of the jointly opcartesians: Using Theorem 8, the jointly opcartesian pairs help us find IPOs.

Definition 26 (Nearly jointly opcartesian). *For $p : \mathbb{E} \to \mathbb{B}$, a cospan f, g is* nearly jointly opcartesian *iff there exists a jointly opcartesian pair f', g' and a vertical ϕ s.t. $f = \phi \circ f'$ and $g = \phi \circ g'$.*

Using that each fibre is a lattice, we find that all cospans are nearly jointly opcartesian.

Proposition 27. *In a predicate sorting $p : \mathbb{E} \to \mathbb{B}$, f, g are jointly opcartesian iff $f = \phi \circ \bar{f}$ and $g = \psi \circ \bar{g}$ where $\bar{f}, \bar{g}$ are opcartesians and ϕ, ψ are the unique verticals given by $\mathrm{cod}(\bar{f}) \sqcup \mathrm{cod}(\bar{g})$.*

Proposition 28. *In a predicate sorting $p : \mathbb{E} \to \mathbb{B}$, every cospan f, g is nearly jointly opcartesian.*

Proof. By Proposition 10, we may write $f = \rho \circ \bar{f}$ and $g = \tau \circ \bar{g}$ where $\bar{f}, \bar{g}$ are opcartesian and ρ, τ are verticals. Take ϕ, ψ to be the unique verticals given by $\bar{f} \sqcup \bar{g}$. By Proposition 27, $\phi \circ \bar{f}$ and $\psi \circ \bar{g}$ are jointly opcartesians, hence there exists a vertical α with $f = \alpha \circ \phi \circ \bar{f}$ and $g = \alpha \circ \psi \circ \bar{g}$. □

Proposition 29. *Let $p : \mathbb{E} \to \mathbb{B}$ be a predicate sorting, and consider a cospan*

$$p(X,Y) \xrightarrow{\ f\ } B \xleftarrow{\ g\ } p(U,V)$$

If f and g have lifts at $p(X,Y)$ and $p(U,V)$, respectively, then they have jointly opcartesian lifts there.

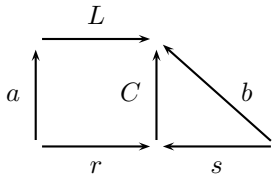
Proof. We have opcartesian lifts $\bar{f}$ of f and $\bar{g}$ of g, and because f, g is a cospan, we may form $\bar{f} \sqcup \bar{g}$; Proposition 27 now gives a jointly opcartesian lift. □

In light of Theorem 8, the above proposition gives us a very tight grip on the relation between IPOs in $\mathbb{B}$ and $\mathbb{E}$. We now use that grip to prove the correspondence theorem.

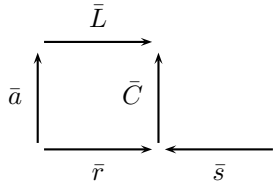
Proof (of Theorem 25). Part 1. Suppose first that $a \; [\![\longmapsto]\!] \; b$. Then for some f, g with $a = p(f)$ and $b = p(g)$, $f \longmapsto g$. Thus $f = D \circ e$ and $g = D \circ e'$ with $(e, e') \in \bar{\mathcal{R}}$, so $a = p(D) \circ p(e)$ and $b = p(D) \circ p(e')$, with $(p(e), p(e')) \in \mathcal{R}$, so $a \longmapsto b$; clearly $P(a)$ and $P(b)$, hence $a \; \lfloor\longmapsto\rfloor \; b$.

Suppose instead that $f \; \lfloor\longmapsto\rfloor \; g$. There exists C, r and s with $(r, s) \in \mathcal{R}$ s.t. $f = C \circ r$ and $g = C \circ s$. By Lemma 21 we can find lifts of r, s at $\bar{\epsilon}$, so by Proposition 29, we have a jointly opcartesian lift $\bar{r}, \bar{s}$ of r, s at ϵ. Again by Lemma 21, we can lift $\overline{C \circ r}$ and $\overline{C \circ s}$ at ϵ; so by $\bar{r}, \bar{s}$ jointly opcartesian, there is a lift $\bar{C}$ of C at $\mathrm{cod}(\bar{r}) = \mathrm{cod}(\bar{s})$. Clearly $(\bar{r}, \bar{s}) \in \bar{R}$, so we have $\bar{C} \circ \bar{r} \longmapsto \bar{C} \circ \bar{s}$, and in turn $f = C \circ r \; [\![\longmapsto]\!] \; C \circ s = g$.

Part 2. Suppose $a \; \lfloor \xrightarrow{L} \rfloor \; b$. Thus there exists $(r, s) \in \mathcal{R}$ and a context C s.t. the following diagram commutes and the square is an IPO.

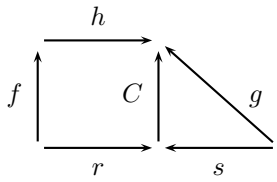


We find $P(C \circ r)$ because $P(L \circ a)$, and $P(C \circ s)$ because $P(b)$. By Lemma 21 and Proposition 19, we have opcartesian lifts $\bar{a}$, $\overline{L \circ a}$, $\bar{r}$, $\bar{s}$, $\overline{C \circ r}$ and $\overline{C \circ s}$ at $\bar{\epsilon}$. Because $\bar{a}$ is opcartesian, we find may a lift of $\bar{L}$ at the codomain of $\bar{a}$; we may assume this lift opcartesian. By Proposition 29, we may assume $\bar{r}, \bar{s}$ jointly opcartesian and $\overline{C \circ r}, \overline{C \circ s}$ cospan, so there exists a lift $\bar{C}$ of C at $\mathrm{cod}(\bar{r}) = \mathrm{cod}(\bar{s})$. Again by Proposition 29, we may assume $\bar{L}, \bar{C}$ jointly opcartesian. Altogether, we have erected the following diagram.



By Theorem 8, we have constructed an IPO, and clearly $(p(\bar{r}), p(\bar{s})) \in \mathcal{R}$, so we have a transition $\bar{a} \xrightarrow{\bar{L}} \bar{C} \circ \bar{s}$. Because $C \circ s = b$, we have obtained the desired transition $a \,[\![\xrightarrow{L}]\!]\, b$.

Suppose instead $a \,[\![\xrightarrow{L}]\!]\, b$. For some f, g, h, we have $a = p(f), b = p(g), L = p(h)$ and $f \xrightarrow{h} g$, so there exists $(r, s) \in \bar{\mathcal{R}}$ and a C s.t. the following diagram commutes, and the square is an IPO.



Clearly, $(p(r), p(s)) \in \mathcal{R}$), and by Theorem 20, the image of the square is an IPO, so there is a transition $p(f) \xrightarrow{p(h)} p(g)$, that is, $a \xrightarrow{L} b$. Clearly, $P(a)$, $P(L)$, $P(b)$ and $P(L \circ a)$, so we have the desired $a \,\lfloor\!\xrightarrow{L}\!\rfloor\, b$. □

7 Context-aware Reactions

Ubiquitous computing is inextricably linked to context-aware computing: computations that are aware of and depend on the present context of the computing agent. Here are two examples. (1) An electronic tour guide device, carried around by visitors at a museum, should provide information about the physically closest exhibit. (2) Doors in a shop which open automatically unless an RFID-tag of an item not registered as sold is too close. Notice the dual requirements in these examples: The first stipulates a positive requirement (the presence of an exhibit), whereas the second stipulates a negative requirement (the absence of an unsold item). Thus, for modeling such applications, it is very convenient if we can specify reaction rules that apply in some but not all contexts. However, as observed in [19], work on process calculi tends to supply at most a rudimentary

distinction between active and passive contexts, a distinction insufficient for the above examples.

We can use sorting to better control reaction: We simply specify our reactive system directly in the sorted category $\mathbb{E}$. By choosing the right codomain for a reaction rule (l, r) we specify in what contexts it applies. In particular, we may use sorting to capture absence of something in the context. In some categories — in particular bigraphs — we model contexts as morphisms and the presence of something as factorization. Thus, we say that $a : A' \to B'$ is present in the context $c : A \to B$ iff $c = x \circ a \circ y$ for some x, y. Under this notion of presence, the predicate sorting can be used to capture *absence* to the extent of the following theorem.

Theorem 30. *Let $p : \mathbb{E} \to \mathbb{B}$ be the predicate sorting, let $f : \epsilon \to B$ be a morphism of $\mathbb{B}$, and let T be any set of morphisms with domain B. Then f has a lift $\bar{f}$ at $\bar{\epsilon}$ s.t. each $g : B \to X$ has a lift at $\mathrm{cod}(\bar{f})$ precisely when $g \in T$ iff T is prefix closed and respects P.*

Put another way: If we want the left-hand side of a reaction rule (l, r) to apply precisely in a set T of contexts, we can do so within any predicate sorting, provided T is prefix-closed and respects the predicate P. Notice that we may take P to be everywhere true, should we so desire.

What does the restriction to prefix-closed sets T mean? Reconsidering the examples with presence and absence, we see that absence is prefix-closed whereas presence is not. Clearly, if a does not occur in a context c, then it also does not occur in any sub-context of c; in particular, it does not occur in any prefix of c. On the other hand, we may very well have a context c that contains some a, but a prefix of c which does not.

In the case of bigraphs or, more generally, wide reactive systems [4, 5], the monoidal structure enables us to express presence without the use of sortings: If we insist that (l, r) applies only when a is present in the context, we simply give the rule as $(a \otimes l, a \otimes r)$. Thus, in sorted wide reactive systems, we can model both presence and absence.

8 Conclusion

Building on earlier work on more specific sortings, ours is the first investigation of general sortings, or type systems, for reactive systems. However, type systems have been investigated for related frameworks, notably for hypergraph rewriting systems in [20], and for process algebras in [21]. Our work is alone in addressing the impact of sorting on labeled transition systems, bisimulations, and congruence properties.

König's typings for hypergraph rewriting systems [20] resembles our sortings in that the aim of typing is explicitly stated to be identifying hypergraphs satisfying a given predicate; that decomposition preserves well-typedness; that composition does not necessarily preserve well-typedness; and that there is a notion of minimal type, roughly comparable to our use of opcartesian lifts. The

method differs from ours — the setting of hypergraphs not withstanding — in that the typing relation is required to satisfy subject reduction, whereas we simply disregard type-altering reductions (cf. the P-restriction of reaction, Definition 24).

Honda's work on typed process algebras [21] is reminiscent of ours in that it focuses explicitly on controlling which morphisms are composible and which are not. However, Honda's notion of process is quite specific to process calculi compared to our more general setting of reactive systems over categories.

For future work, we see as the most urgent the reconciliation of present work with precategories, bridging the gap to bigraphs. Observing that our notion of sorting applies immediately to abstract bigraph, we are hopeful that we can transfer our results across the quotient functors to precategories.

Other directions include further investigating compositionality of sortings. In Section 3, we demonstrated how to compose sortings sequentially and how to form their conjunction; it is natural to wonder about other connectives, particularly negation. Another direction is investigating the use of sortings for encoding typed calculi in reactive systems. Yet another is that for bigraphs, it would be interesting if there were stronger connections between BiLog [17, 18] and sorted bigraphs than those noted in Section 4. For instance, BiLog formulas might form the basis of a syntactic formulation of sorting, which could in turn be useful for implementations of reactive systems. Finally, it would be interesting to know if the predicate sorting is in some sense universal.

Acknowledgments. We gratefully acknowledge good suggestions from the anonymous referees and vibrant discussions with Rasmus Lerchedahl Petersen and Mikkel Bundgaard.

References

1. Sewell, P.: From rewrite rules to bisimulation congruences. Theoretical Computer Science **274**(1–2) (2002) 183–230
2. Leifer, J.J., Milner, R.: Deriving bisimulation congruences for reactive systems. In: CONCUR '00: Proceedings of the 11th International Conference on Concurrency Theory, Springer-Verlag (2000) 243–258
3. Jensen, O.H., Milner, R.: Bigraphs and transitions. In: POPL '03: Proceedings of the 30th ACM SIGPLAN-SIGACT symposium on Principles of programming languages, ACM Press (2003) 38–49
4. Jensen, O.H., Milner, R.: Bigraphs and mobile processes (revised). Technical Report UCAM-CL-TR-580, University of Cambridge Computer Laboratory (2004)
5. Milner, R.: Pure bigraphs: Structure and dynamics. Information and Computation **204**(1) (2006) 60–122
6. Sassone, V., Sobocinski, P.: Deriving bisimulation congruences: 2-categories vs. precategories. In: FOSSACS '03: Proceedings of Foundations of Software Science and Computation Structures. Volume 2620 of Lecture Notes in Computer Science., Springer-Verlag (2003) 409–424
7. Sassone, V., Sobocinski, P.: Reactive systems over cospans. In: LICS '05: Proceedings of the twentieth annual IEEE symposium on Logic in computer science, IEEE Computer Society Press (2005) 311–320

8. Bruni, R., Gadducci, F., Montanari, U., Sobociński, P.: Deriving weak bisimulation congruences from reduction systems. In: CONCUR '05: Proceedings of the 16th international conference on Concurrency theory. Volume 3653 of Lecture Notes in Computer Science., Springer-Verlag (2005) 293–307
9. Jensen, O.H.: Mobile Processes in Bigraphs. PhD thesis, University of Aalborg (2006) Forthcoming.
10. Milner, R., Leifer, J.J.: Transition systems, link graphs and Petri nets. Technical report, University of Cambridge, Computer Laboratory (2004)
11. Bundgaard, M., Hildebrandt, T.: Bigraphical semantics of higher-order mobile embedded resources with local names. In: GT-VC '05: Proceedings of the Graph Transformation for Verification and Concurrency workshop. Volume 154 of Electronic Notes in Theoretical Computer Science., Elsevier (2006) 7–29
12. Birkedal, L., Debois, S., Elsborg, E., Hildebrandt, T., Niss, H.: Bigraphical Models of Context-aware Systems. Technical Report 74, IT University of Copenhagen (2005) ISBN: 87-7949-110-3.
13. Birkedal, L., Debois, S., Elsborg, E., Hildebrandt, T., Niss, H.: Bigraphical Models of Context-aware Systems. In: FOSSACS '06: Proceedings of 9th International Conference on Foundations of Software Science and Computation Structures. Volume 3921 of Lecture Notes in Computer Science., Springer-Verlag (2006)
14. Birkedal, L., Debois, S., Hildebrandt, T.: Sortings for reactive systems. Technical Report 84, IT University of Copenhagen (2006) ISBN 87-7949-124-3.
15. Jacobs, B.: Categorical logic and type theory. Volume 141 of Studies in logic and the foundation of mathematics. Elsevier (1999)
16. Poincaré, H.: Science and Method. Dover Publications (1914) Translation by Francis Maitland.
17. Conforti, G., Macedonio, D., Sassone, V.: Spatial logics for bigraphs. In: ICALP '05: Proceedings of the 32nd international colloquium on Automata, languages and programming. Volume 3580 of Lecture Notes in Computer Science., Springer-Verlag (2005) 766–778
18. Conforti, G., Macedonio, D., Sassone, V.: Spatial logics for bigraphs. Computer Science Report 02, University of Sussex (2005)
19. Braione, P., Picco, G.P.: On calculi for context-aware coordination. In: COORDINATION '04: Proceedings of the 7th International Conference on Coordination Models and Languages. Volume 2949 of Lecture Notes in Computer Science., Springer (2004) 38–54
20. König, B.: A general framework for types in graph rewriting. In: FST TCS '00: Proceedings of the 20th Conference on Foundations of software technology and Theoretical computer science. Volume 1974 of Lecture Notes in Computer Science., Springer-Verlag (2000) 373–384
21. Honda, K.: Composing processes. In: POPL '96: Proceedings of the 23rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages. (1996) 344–357