Pure bigraphs: structure and dynamics

Robin Milner

University of Cambridge Computer Laboratory, Cambridge CB3 0FD, UK

Abstract

Bigraphs are graphs whose nodes may be nested, representing locality, independently of the edges connecting them. They may be equipped with reaction rules, forming a *bigraphical reactive system* (Brs) in which bigraphs can reconfigure themselves. Following an earlier paper describing *link graphs*, a constituent of bigraphs, this paper is a devoted to *pure* bigraphs, which in turn underlie various more refined forms. Elsewhere it is shown that behavioural analysis for Petri nets, π-calculus and mobile ambients can all be recovered in the uniform framework of bigraphs.

The paper first develops the dynamic theory of an abstract structure, a *wide reactive system* (Wrs), of which a Brs is an instance. In this context, labelled transitions are defined in such a way that the induced bisimilarity is a congruence. This work is then specialised to Brss, whose graphical structure allows many refinements of the theory. The latter part of the paper emphasizes bigraphical theory that is relevant to the treatment of dynamics via labelled transitions. As a running example, the theory is applied to finite pure CCS, whose resulting transition system and bisimilarity are analysed in detail.

The paper also mentions briefly the use of bigraphs to model pervasive computing and biological systems.

Part I : Mathematical Framework

The introduction provides a rationale for bigraphs, and a synopsis of the whole paper. Section 2 introduces *s-categories*, including the notion of *support* which will be used to identify occurrences of entities within bigraphs; it also defines *relative pushouts* (RPOs), which are important for the behavioural theory of bigraphs. Section 3 introduces the abstract notion of a *wide reactive system* (Wrs); it is not graphical, but gives prominence to spatial extension, or *width*. Section 4 defines *transition systems* for Wrss, which may be used to define bisimilarities and other behavioural relations. It is shown that, for transitions based on RPOs, bisimilarity is a congruence. Varieties of transition system are defined and analysed.

1 Introduction

Bigraphical reactive systems [33,35,34,22,23,36,27] are a graphical model of computation in which both *locality* and *connectivity* are prominent. Recognising the increasingly topographical quality of global computing, they take up the challenge to base all distributed computation on topographical structure. A typical bigraph is shown in Figure 1; it represents a highly simplified system of information flow and computation in a built environment, which we shall soon discuss in more detail. Such a graph is reconfigurable, and its nodes (the ovals and circles) may represent a great variety of computational objects: a physical location, an administrative region, a human agent, a mobile phone, a computer, a sensor, a data constructor, a π-calculus input guard, a mobile ambient, a cryptographic key, a message, a replicator, and so on.

Bigraphs are a development of action calculi [32], but simpler. They use ideas from many sources: the Chemical Abstract machine (Cham) of Berry and Boudol [3], the π-calculus of Milner, Parrow and Walker [40], the interaction nets of Lafont [24], the mobile ambients of Cardelli and Gordon [8], the explicit fusions of Gardner and Wischik [18] developed from the fusion calculus of Parrow and Victor [42], Nomadic Pict by Wojciechowski and Sewell [52], and the uniform approach to a behavioural theory for reactive systems of Leifer and Milner [26].

The mathematical structure of bigraphs allows concepts to be treated somewhat independently. For example, *connectivity* and *locality* are treated orthogonally. This paper is the second in a series which present bigraphs incrementally, with suitable examples at each stage. The first in the series [27] presented *link graphs*, which have connectivity but lack locality; the main illustration was a study of Petri nets with an associated transition system. The current paper adds locality, orthogonal to connectivity; the term *pure bigraphs* means that locality does not yet support a notion of scope, or binding of names. A study of finite pure CCS, together with a derived transition system for it, forms a running example. This paper us self-contained, but a familiarity with the previous paper will be helpful.

Refinements of pure bigraphs, including *binding bigraphs* [23], can typically derive much of their theory from pure bigraphs; they will be treated in later publications.

The challenge from applications

The long-term aim of this work is to model computation on a global scale, as represented by the Internet and the Worldwide Web, and more recently by pervasive computing. The aim is not just to model systems already designed and running; beyond that, we seek a theory to guide the specification and programming of these systems, and to guide their future adaptation. The so-called *vanishing ubiquitous computer* of the future is within reach of today's technology. To *understand* it is a goal less widely perceived, but nonetheless essential if we are to avoid the systems that are as stagnant and inscrutable as today's legacy software, and on an even larger scale.

So we have to reverse the typical order of events in which design and implementation come first, modelling later. (For example, programming languages are hardly ever based thoroughly on a theoretical model, yet they are pivotal in all our implementations.) Such 'retro-modelling' leads to an understanding of designed systems that is brittle, because it deteriorates seriously as the systems evolve under changing demand. In the long run, system designs must be expressed from the outset with the concepts and notations of a theory rich enough to encompass all that the designers wish.

The arrival of ubiquitous mobile computing offers an opportunity for this, simply because it is new enough for its languages and implementation techniques not to be entrenched. Moreover, concurrency theories already provide a conceptual frame in which to study distributed mobile systems, and they offer structures for new languages. Thus, through experimental applications, designers and analysts may come to speak the same tongue. As a specialised but significant example, both Petri nets and the π-calculus are now adopted to assist design of systems for the management of business processes [50].

Global computing presents huge demands, and we cannot expect to arrive immediately at the right model for it. We have to strike a compromise between fine-tuning existing models on the one hand, and making too large a leap on the other hand. A model must grasp many aspects of real systems if it is to be seriously used in experimental design, and thus provide the feedback necessary to improve the model itself. If we merely adopt the classic scientific approach of tackling each aspect of global computing separately, we may develop elegant separate theories yet find ourselves unable to unify them. On the other hand to tackle all aspects is too hard. This uncomfortable dilemma is not faced in natural science, since there the objects of

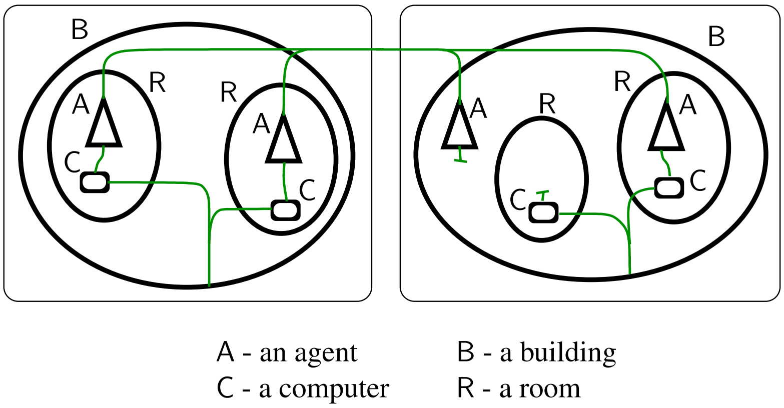


Fig. 1. A bigraph for communication in a built environment

study typically remain stable —in so far as they are independent of human designs.

Our strategy here is to tackle just two aspects of mobile systems simultaneously: *mobile locality* and *mobile connectivity*. Already this combination presents a challenge: to what extent are locality and connectivity interdependent? In plain words, does *where you are* affect *whom you can talk to*? The answer must lie in the level of modelling. To a user of the Internet (seeing it abstractly) there is total independence, and we want to model it at a high level, just as it appears to users. But to the engineer these remote communications are not atomic; they involve chains of interactions between neighbouring entities, and we must also provide a low-level model which reflects this reality. These two levels must surely be part of a single multi-level model that explains how higher levels are *realised* by lower levels.

Of the two levels, the lower is the less novel. Indeed, von Neumann's cellular automata are the original paradigm for it; his agents were arranged on a fixed grid and interaction could only occur between neighbours. But in such a concrete model we hope to *realise* a higher level view in which a single agent is represented by different cells at different moments, and may send messages to other distant agents. So the challenge we address here is to provide the means to view locality and connectivity as dependently —or independently— as you wish, and to correlate these views. This seems to require new mathematical structures, and bigraphs attempt to provide them.

Example 1 (sentient buildings) As a simple illustration, consider a crude version of a *sentient built environment*, modelled as a bigraph in Figure 1. There are two structures on the nodes of a bigraph; they may be nested, and they may also be connected by links. The linkage is independent of the nesting, so links often cross node boundaries. Nodes may be of many kinds, each represented by a *control* (A, B, ...) associated with each node. (The shape of nodes is suggestive but redundant.) For this particular bigraph:

- The two regions (large squares), each with one building (B), may lie arbitrarily far apart in a larger system, e.g. one in France and one in Australia.

- The four agents (A), perhaps humans equipped with devices, are conducting a conference call.
- The computers (C) in each building are networked as part of its infrastructure —another embedded subsystem.
- Many reconfigurations are possible. An agent may abandon the conference call; an agent may enter or leave a room (R); on entry, the computer (equipped with sensor) may connect with him/her; a computer network may contribute to the conference call; a room may become inoperative because of fire; and so on. ■

We have so far considered only discrete events; but continuous events and stochastic behaviour must also be modelled. These structures for modelling such man-made systems are not far from those (discussed later) that have already been used to model behaviour of biological cells.

In defining bigraphs for such modelling, we wish to embrace familiar calculi of mobile processes, which deal with interaction and mobility in different ways. We also want a theory that can be specialised to each of these calculi, and therefore unifies them. This leads naturally to the second of our twin challenges.

The challenge from process calculi

Existing process calculi have made great progress with communication [5,2,20,30], mobile connectivity [40,15] and mobile locality [3,8]. There is some agreement among them, and their behavioural theories are well developed. At the same time the space of possible calculi is large and not well understood. In particular, as shown by Castellani's [9] comprehensive survey, widely varying notions of locality have been explored; this implies equal variety in their treatment of mobility.

The challenge from process calculi is to provide a uniform behavioural theory, so that many process calculi can be expressed in the same frame without seriously affecting their treatment of behaviour. It is important to assess clearly the value of a uniform approach, in order to avoid superficial claims. For work involving the *application* of an established process calculus, encoding into a common framework such as bigraphs is unlikely to yield much benefit. But for assessing the *theory* of an existing calculus there will be benefits; as an example, the study in Jensen's forthcoming dissertation [21] throws light upon the various bisimilarities for π-calculus, and explains weak bisimilarity (where silent transitions are ignored) in terms of a more general phenomenon. Finally in defining new process calculi, which may well be needed for pervasive computing, a uniform framework will avoid adhoc repetitive theory and could provide immediate insights for designing the calculus.

We now outline how research leading up to the bigraphical model has addressed this challenge.

It is common to present the *dynamics* of processes by means of *reactions* (also known as rewriting rules) of the form $r \longrightarrow r'$, meaning that r can change its state to r' in suitable contexts. In process calculi this treatment is typically refined into *labelled transitions* of the form $a \xrightarrow{\ell} a'$, where the label ℓ is drawn from some vocabulary expressing the possible interactions between an agent a and its environment. These transitions have the great advantage that they support the definition of behavioural preorders and equivalences, such as traces, failures and bisimilarity. But the definition of those transitions tends to be tailored for each calculus.

So can these labels be *derived* uniformly, given a set of reaction rules of the form $r \longrightarrow r'$? A natural approach is to take the labels to be certain (environmental) *contexts*; if L is such a context, the transition $a \xrightarrow{L} a'$ implies that a reaction can occur in $L \circ a$ leading to a new state a'. (As we shall see, bigraphical agents and contexts live in a category, or more generally an s-category, where the composition $L \circ a$ represents the insertion of agent a in context L.) Moreover, we would like to be sure that the behavioural relations —such as bisimilarity— that are determined by the transitions are well-behaved.

But we don't want *all* contexts as labels; as Sewell [49] points out, the behavioural equivalences that result from this choice are unsatisfactory. How to find a satisfactory —and suitably minimal— set of labels, and to do it uniformly, remained open for many years. As a first step, Sewell was able uniformly to derive satisfactory context-labelled transitions for parametric term-rewriting systems with parallel composition and blocking, and showed bisimilarity to be a congruence. It remained a problem to do it for reactive systems dealing with connectivity, such as the π-calculus.

This was overcome by Leifer and Milner [26], who defined minimal labels in terms of the categorical notion of *relative pushout* (RPO), also ensuring that behavioural equivalence is a congruence. These results were extended and refined in Leifer's PhD Dissertation [25], and applied to action graphs with rich connectivity [10]. Meanwhile bigraphs were developed from action graphs; they were inspired by the simplicity that comes from treating locality and connectivity independently, by the mobile ambients of Cardelli and Gordon, and by Gardner's development [17] of action graphs with undirected edges. This theoretical development has been augmented by a sequence of case studies applying the bigraph model to existing calculi, including the π-calculus [22,23], mobile ambients [21], Petri nets [36,27] and the λ-calculus [38]. These give confidence that the model can incorporate existing theories.

Each of the case studies involved some specialisation of the bigraph model. The present paper is devoted to *pure bigraphs*, which underlie these specialisations. It concentrates mainly upon the theory but illustrates it by application to finite pure CCS as a running example. Sequel papers will specialise the model in various ways, for example to *binding bigraphs* which allow scope and binding for certain names,

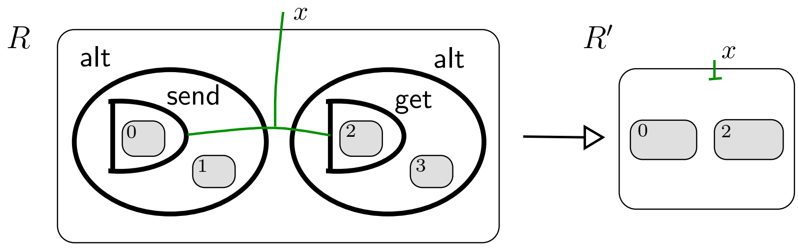


Fig. 2. A bigraphical reaction rule for CCS with summation

thus admitting more refined applications. It will be seen that the basic theory of pure bigraphs is preserved by these specialisations, thus establishing pure bigraphs as a core theory.

However, the theory cannot claim to be definitive; many variations are possible. Therefore this work has been divided as much as possible into separate topics, making it more amenable to variation. For example, *bigraphs* themselves are defined in terms of two independent structures, *place graphs* and *link graphs*, and each of these can be varied. Also, *bigraphical reactive systems* (Brss) are defined as merely one instance of a general concept, *wide reactive systems* (Wrss), whose abstract theory we develop in Part I; many other instances are possible.

We now introduce our running example.

Example 2 (reaction in CCS) The calculus CCS [30] has a reaction rule

$$(\overline{x}.P + M) \,|\, (x.Q + N) \longrightarrow P \,|\, Q \;,$$

where $\overline{x}.P$ and $x.Q$ are guarded output and input respectively, while M and N represent zero or more alternatives of the same nature. The rule represents a communication on channel x, which may preempt other possible communicators on the same channel; the result of the communication is to allow the continuations P and Q to continue in parallel, while the alternatives M and N are discarded.

Figure 2 shows the corresponding reaction rule in bigraphs. It uses three controls: send for output, get for input and alt for alternation. They are declared to be *passive* controls, i.e. no reaction can occur inside them. The reaction rule means that the *redex* R occurring in a larger bigraph, with anything in its holes (grey boxes), can be replaced by the *reactum* R', retaining some of the contents of R as indicated by the ordinals in its holes. Note several points:

- The send- and get- nodes are connected in R by a link named x. In the larger context these may be linked to competitors for communication on that link. Nothing in R' retains that link, but competitors in the larger context will retain it.
- The occupants of the holes —collectively called the *parameter* of the reaction— may freely be linked to the larger context (and to each other); they may even contain uses of the link x, which may later be activated.

- Bigraphs are rigorous entities. Besides their diagrams, they may be written and manipulated algebraically. Here is the algebraic form of the reaction rule, mildly sugared to clarify which hole is which:

$$\mathsf{alt}.\,(\mathsf{send}_x.\square_0 \mid \square_1) \mid \mathsf{alt}.\,(\mathsf{get}_x.\square_2 \mid \square_3) \longrightarrow x \mid \square_0 \mid \square_2 \ .$$

  The forms such as $\mathsf{alt}.\,(\ldots)$ and $\mathsf{send}_x.\square_0$, are derived from categorical compositions; the parallel combinator '$\mid$' is derived from tensor product.
- Those familiar with CCS may be surprised that no special bigraph combinator is needed to encode the summation '$+$'. Instead, a control alt does the job, together with the parallel combinator '$\mid$'. The latter combinator is purely *structural*, placing bigraphs side-by-side; the *dynamic* behaviour of CCS summation — for example, that only one summand of a sum will be allowed to execute — is captured in bigraphs by the reaction rule and the passivity of the control alt.

We shall return to this example from time to time in the following sections, to illustrate various points. In Section 11 we shall encode finite CCS into bigraphs, and illustrate our uniformly derived strong bisimilarity by showing that it exactly captures the one originally defined for CCS. ∎

Synopsis

The paper's three parts play distinct roles. Each Part begins with an abstract, but the following brief overview will be helpful.

Part I is entirely devoted to a mathematical framework consisting of s-categories and a way of providing them with dynamics; in this framework, many other models beside bigraphs can be set up. The purpose is to develop theory that will apply to future enrichments and variations of the bigraph model.

Part II is entirely concerned with the static structure of bigraphs. The mere definition of bigraphs is not complex, but it admits a large taxonomy and many operations; the emphasis in this Part is to identify elementary bigraphs from which others can be built, as well as basic operations from which others can be derived. It is also shown how the *static* theory of Part I is instantiated in bigraphs.

Part III establishes the dynamic theory of bigraphs, and shows in turn how the *dynamic* theory of Part I is instantiated. This leads to further taxonomy, some refinements of the theory, and in particular a notion of *sorting*; all of these are then applied to recover some of the original theory of CCS. Finally, the concluding section points to related research and future directions.

Acknowledgement I am greatly indebted to Ole Høgh Jensen and Jamey Leifer, whose ideas have been important in developing bigraphs. I would also like to thank the anonymous referees for comments which have helped me to improved the paper.

2 S-categories and relative pushouts

In this section and the following one we develop a mathematical framework for the static and dynamic properties of bigraphs. There are several varieties of bigraph, and in this general setting we shall derive properties that will apply to all of them. This work relies substantially on results on Leifer's PhD dissertation [25], and has been presented in [27]; it is repeated briefly here to make the present paper self-contained.

Notation We accent the name of a precategory or s-category, as in $\acute{\mathbf{C}}$, to distinguish it from a category. We use $I, J, K, \ldots$ for objects and $f, g, h, \ldots$ for arrows. We use juxtaposition for composition, 'id' for identity and '$\otimes$' and tensor product. We denote the domain I and codomain J of an arrow $f \colon I \to J$ by $\mathsf{dom}(f)$ and $\mathsf{cod}(f)$; the set of arrows from I to J, called a *homset*, is denoted by $\acute{\mathbf{C}}(I \to J)$. ■

Id_S will denote the identity function on a set S, and $\emptyset_S$ the empty function from $\emptyset$ to S. We shall use $S \uplus T$ for union of sets S and T known or assumed to be disjoint, and $f \uplus g$ for union of functions whose domains are known or assumed to be disjoint. This use of $\uplus$ on sets should not be confused with the disjoint sum '$+$', which disjoins sets *before* taking their union. We assume a fixed representation of disjoint sums; for example, $X + P + Y$ means $(\{0\} \times X) \cup (\{1\} \times P) \cup (\{2\} \times Y)$, and $\sum_{v \in V} P_v$ means $\bigcup_{v \in V} (\{v\} \times P_v)$. We write $f \restriction S$ for the restriction of a function f to the domain S, and $R \restriction S$ for the restricted relation $R \cap S^2$.

A natural number m is often interpreted as a finite ordinal $m = \{0, 1, \ldots, m-1\}$. We often use i to range over m; when $m = 2$ we use $\bar{\imath}$ for the complement $1 - i$ of i. We use $\vec{x}$ for a finite sequence $\{x_i \mid i \in m\}$; when $m = 2$ this is an ordered pair.

Definition 2.1 (precategory) A *precategory* $\acute{\mathbf{C}}$ is defined exactly as a category, except that the composition of arrows is not always defined. Composition with the identities is always defined, and $\mathsf{id}\, f = f = f\, \mathsf{id}$. In the equation $h(gf) = (hg)f$, the two sides are either equal or both undefined. ■

We shall extend categorical concepts to precategories without comment when they are unambiguous. We now extend explicitly the concept of monoidal category:

Definition 2.2 (tensor product, monoidal precategory) A *strict, symmetric monoidal* precategory has a partial *tensor product* $\otimes$ both on objects and on arrows. It has a unit object ϵ, called the *origin*, such that $I \otimes \epsilon = \epsilon \otimes I = I$ for all I. Given $I \otimes J$ and $J \otimes I$ it also has a *symmetry* isomorphism $\gamma_{I,J} : I \otimes J \to J \otimes I$. The tensor and symmetries satisfy the following equations when both sides exist:

$$(1)\;\; f\otimes(g\otimes h)=(f\otimes g)\otimes h \;\text{ and }\; \mathsf{id}_\epsilon\otimes f=f \qquad (3)\;\; \gamma_{I,\epsilon}=\mathsf{id}_I$$

$$(2)\;\; (f_1\otimes g_1)(f_0\otimes g_0)=f_1f_0\otimes g_1g_0 \qquad (4)\;\; \gamma_{J,I}\gamma_{I,J}=\mathsf{id}_{I\otimes J}$$

$$(5)\;\; \gamma_{I,K}(f\otimes g)=(g\otimes f)\gamma_{H,J} \quad (\text{for } f: H\to I,\; g: J\to K)$$

$$(6)\;\; \gamma_{I\otimes J,K}=(\gamma_{I,K}\otimes\mathsf{id}_J)(\mathsf{id}_I\otimes\gamma_{J,K})\,. \qquad \blacksquare$$

'Strict' means that condition 1 holds exactly, not merely up to isomorphism; 'symmetric' refers to the symmetry isomorphisms satisfying conditions 3–6.

In this work we shall use *s-categories*, a well-behaved form of precategory. A particular case will be when arrows are bigraphs. For the present, think of these as ordinary graphs. In our dynamic theory we shall need to make explicit the extent to which two graphs share nodes, when they occur as subgraphs of a larger graph. One way this can be achieved in a categorical framework is to work with graphs as objects, and embeddings as arrows. But there is a gain in treating the *graphs* as arrows (between *interfaces* as objects); for then the categorical composition and tensor product are ways of building larger graphs from smaller ones.

In s-categories, each arrow is equipped with a set called its *support*. The support of a bigraph will include its nodes, and this immediately allows us to handle occurrences and the sharing of nodes. This is not mere convenience; our dynamic theory of bigraphs depends strongly on the notion of a *relative pushout* (RPO), defined later in this section. If we model bigraphs as arrows in an ordinary category, RPOs do not exist (a counter-example, Example 6, is shown in Section 6). In Part II we show how to construct RPOs for bigraphs in s-categories.

We discuss alternative approaches in Section 12.

Definition 2.3 (s-category) An *s-category* $\acute{\mathbf{C}}$ is a strict symmetric monoidal precategory which has:

- for each arrow f, a finite set $|f|$ called its *support*, such that $|\mathsf{id}_I| = \emptyset$. For $f\colon I\to J$ and $g\colon J\to K$ the composition $gf\colon I\to K$ is defined iff $|g|\cap|f| = \emptyset$ and $\mathsf{dom}(g) = \mathsf{cod}(f)$; then $|gf| = |g|\uplus|f|$. Similarly, for $f\colon H\to I$ and $g\colon J\to K$ with $H\otimes J$ and $I\otimes K$ defined, the tensor product $f\otimes g\colon H\otimes J\to I\otimes K$ is defined iff $|f|\cap|g|=\emptyset$; then $|f\otimes g| = |f|\uplus|g|$.
- for any arrow $f : I\to J$ and any injective map ρ whose domain includes $|f|$, an

arrow $\rho\bullet f : I \to J$ called a *support translation* of f such that

$$\begin{array}{ll}
(1)\ \ \rho\bullet\mathsf{id}_I = \mathsf{id}_I & (4)\ \ \mathsf{Id}_{|f|}\bullet f = f \\
(2)\ \ \rho\bullet(gf) = (\rho\bullet g)(\rho\bullet f) & (5)\ \ (\rho_1 \circ \rho_0)\bullet f = \rho_1\bullet(\rho_0\bullet f) \\
(3)\ \ \rho\bullet(f \otimes g) = \rho\bullet f \otimes \rho\bullet g & (6)\ \ \rho\bullet f = (\rho \upharpoonright |f|)\bullet f \\
 & (7)\ \ |\rho\bullet f| = \rho(|f|)\ .
\end{array}$$

Each equation is required to hold only when both sides are defined. ∎

We now consider functors between s-categories.

Definition 2.4 (support equivalence, supported functor) Two arrows $f, g : I \to J$ in an s-category $\acute{\mathbf{C}}$ are *support-equivalent*, written $f \doteq g$, if $\rho\bullet f = g$ for some support translation ρ. By Definition 2.3 this is an equivalence relation. If $\acute{\mathbf{D}}$ is another s-category, then a *functor* $\mathcal{F}\colon \acute{\mathbf{C}} \to \acute{\mathbf{D}}$ is a function on objects and arrows that preserves identities, composition, tensor product and support equivalence. If $\mathcal{F}$ is an inclusion function then $\acute{\mathbf{C}}$ is a *sub-s-category* of $\acute{\mathbf{D}}$. ∎

When we no longer need to keep track of support we may use a quotient *category* (not just s-category). To define such quotients, we need the following notion:

Definition 2.5 (congruence) Let $\equiv$ be an equivalence defined on every homset of a supported precategory $\acute{\mathbf{C}}$. We say that $\equiv$ is *preserved* by an operator $*$ if $f \equiv f'$ and $g \equiv g'$ imply $f * g \equiv f' * g'$ whenever the latter are defined. Then $\equiv$ is *congruence on* $\acute{\mathbf{C}}$ whenever it is preserved by composition and tensor product. ∎

As an example of a simple congruence on bigraphs, we may define $f \equiv f'$ to mean that f and f' are identical when all K-nodes are discarded, for some particular control K. The most important example of a congruence will be support equivalence ($\doteq$). The following definition shows that any congruence at least as coarse as support equivalence will yield a well-defined quotient category:

Definition 2.6 (quotient categories) Let $\acute{\mathbf{C}}$ be a supported precategory, and let $\equiv$ be a congruence on $\acute{\mathbf{C}}$ that includes support equivalence, i.e. $\doteq\ \subseteq\ \equiv$. Then the *quotient* of $\acute{\mathbf{C}}$ by $\equiv$ is a category $\mathbf{C} \stackrel{\text{def}}{=} \acute{\mathbf{C}}/\!\equiv$, whose objects are the objects of $\acute{\mathbf{C}}$ and whose arrows are equivalence classes of arrows in $\acute{\mathbf{C}}$:

$$\mathbf{C}(I, J) \stackrel{\text{def}}{=} \{\, [f]_{\equiv} \mid f \in \acute{\mathbf{C}}(I, J) \,\}\ .$$

In $\mathbf{C}$, the identities, composition and tensor product are given by

$$\begin{aligned} \mathsf{id}_m &\stackrel{\text{def}}{=} [\mathsf{id}_m]_\equiv \\ [g]_\equiv [f]_\equiv &\stackrel{\text{def}}{=} [gf]_\equiv \\ [f]_\equiv \otimes [g]_\equiv &\stackrel{\text{def}}{=} [f \otimes g]_\equiv \ . \end{aligned}$$

By assigning empty support to every arrow we may also regard $\mathbf{C}$ as an s-category, and we call $[\cdot]_\equiv : \acute{\mathbf{C}} \to \mathbf{C}$ the $\equiv$-*quotient functor* for $\acute{\mathbf{C}}$. ■

Note that the quotient does not affect objects; thus any tensor product on $\mathbf{C}$ may still be partial on objects. But $\mathbf{C}$ is indeed a category; composition is always well-defined because $f \doteq g$ implies $f \equiv g$, and so also is tensor product provided it is defined on the domains and codomains. We often abbreviate $[\cdot]_\doteq$ to $[\cdot]$; we call it the *support quotient functor*. From the definition, clearly a coarser quotient $[\cdot]_\equiv$ exists whenever $\equiv$ is a congruence that includes support equivalence.

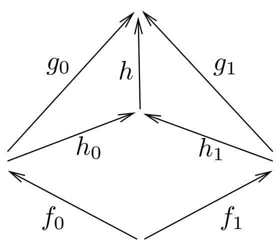
We now turn to the notion of relative pushout (RPO), which will be of crucial importance in defining labelled transitions. The rest of this section, except where stated, pertains to any precategory.

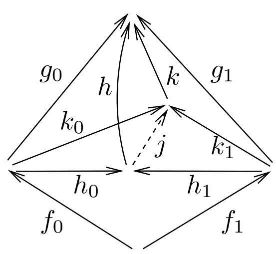
Notation We shall often use $\vec{f}$ for a pair f_0, f_1 of arrows; similarly for objects. For example, if the two arrows share a domain H and have codomains I_0, I_1 we write $\vec{f} : H \to \vec{I}$. ■

Definition 2.7 (bound, consistent) If two arrows $\vec{f} : H \to \vec{I}$ share domain H, the pair $\vec{g} : \vec{I} \to K$ share codomain K and $g_0 f_0 = g_1 f_1$, then we say that $\vec{g}$ is a *bound* for $\vec{f}$. If $\vec{f}$ has any bound, then it is said to be *consistent*. ■

Definition 2.8 (relative pushout) Let $\vec{g} : \vec{I} \to K$ be a bound for $\vec{f} : H \to \vec{I}$. A *bound for* $\vec{f}$ *relative to* $\vec{g}$ is a triple $(\vec{h}, h)$ of arrows such that $\vec{h}$ is a bound for $\vec{f}$ and $hh_i = g_i$ ($i = 0, 1$). We may call the triple a *relative bound* when $\vec{g}$ is understood.

A *relative pushout* (*RPO*) *for* $\vec{f}$ *relative to* $\vec{g}$ is a relative bound $(\vec{h}, h)$ such that for any other relative bound $(\vec{k}, k)$ there is a unique arrow j for which $jh_i = k_i$ ($i = 0, 1$) and $kj = h$. ■





We shall often omit the word 'relative'; for example we may call $(\vec{h}, h)$ a bound (or

RPO) for $\vec{f}$ to $\vec{g}$.

The more familiar notion, a pushout, is a bound $\vec{h}$ for $\vec{f}$ such that *for any* bound $\vec{g}$ there exists an h which makes the left-hand diagram commute. Thus a pushout is a *least* bound, while an RPO provides a *minimal* bound relative to a given bound $\vec{g}$. In bigraphs we shall find that RPOs exist in cases where there is no pushout; see Constructions 6.8 and the discussion preceding it.

Now suppose that we can create an RPO $(\vec{h}, h)$ for $\vec{f}$ to $\vec{g}$; what happens if we try to iterate the construction? More precisely, is there an RPO for $\vec{f}$ to $\vec{h}$? The answer lies in the following important concept:

Definition 2.9 (idem pushout) A pair $\vec{h} : \vec{I} \to J$ is an *idem pushout* (*IPO*) for the pair $\vec{f} : H \to \vec{I}$ if the triple $(\vec{h}, \mathsf{id}_J)$ is an RPO for $\vec{f}$ to $\vec{h}$. ■

Then it turns out that the attempt to iterate the RPO construction will yield the *same* bound (up to isomorphism); intuitively, the minimal bound for $\vec{f}$ to any bound $\vec{g}$ is reached in just one step. This is a consequence of the first two parts of the following proposition, which summarises the essential properties of RPOs and IPOs on which our work relies. They are proved for categories in Leifer's Dissertation [25] (see also Leifer and Milner [26]), and the proofs can be routinely adapted for precategories.

Proposition 2.10 (properties of RPOs) *In any precategory* $\acute{\mathbf{C}}$*:*

(1) If an RPO for $\vec{f}$ to $\vec{g}$ exists, then it is unique up to isomorphism.
(2) If $(\vec{h}, h)$ is an RPO for $\vec{f}$ to $\vec{g}$, then $\vec{h}$ is an IPO for $\vec{f}$.
(3) If $\vec{h}$ is an IPO for $\vec{f}$, and an RPO exists for $\vec{f}$ to hh_0, hh_1, then the triple $(\vec{h}, h)$ is such an RPO.
(4) (IPO pasting) Suppose that the diagram below commutes, and that f_0, g_0 has an RPO to the pair h_1h_0, f_2g_1. Then
(a) if the two squares are IPOs, so is the big rectangle;
(b) if the big rectangle and the left square are IPOs, so is the right square.

$$\begin{tikzcd} \cdot \arrow[r, "h_0"] & \cdot \arrow[r, "h_1"] & \cdot \\ \cdot \arrow[u, "f_0"] \arrow[r, "g_0"'] & \cdot \arrow[u, "f_1"] \arrow[r, "g_1"'] & \cdot \arrow[u, "f_2"'] \end{tikzcd}$$

(5) (IPO sliding) If $\acute{\mathbf{C}}$ is an s-category then IPOs are preserved by support translation; that is, if $\vec{g}$ is an IPO for $\vec{f}$ and ρ is any injective map whose domain includes the supports of $\vec{f}$ and $\vec{g}$, then $\rho \bullet \vec{g}$ is an IPO for $\rho \bullet \vec{f}$.

3 Wide reactive systems

We now introduce a kind of dynamical system, of which bigraphs will be an instance. In previous work [26,**?**] a notion of reactive system was defined. In our present terms, this consisted of an s-category whose arrows are called *contexts*, including *agents* whose domain is the origin ϵ, together with a set of agent-pairs (r, r') called *reaction rules*, and a sub-s-category of so-called *active* contexts. The reaction relation $\longrightarrow\!\!\!\!\triangleright$ between agents was taken to be the smallest such that $Dr \longrightarrow\!\!\!\!\triangleright Dr'$ for every active context D and reaction rule (r, r').

For such systems, labelled transitions based upon IPOs have been derived uniformly [26]. Several behavioural preorders and equivalences based upon these transitions —including bisimilarity— were shown to be congruences, subject to two conditions: first, that sufficient RPOs exist in the precategory; second, that decomposition preserves activity — i.e. DC active implies both C and D active.

In subsequent work, RPOs were found in interesting cases (Leifer [25], Cattani et al [10]). Each case met the condition that decomposition preserves activity, if we limit attention to contexts with a single hole. However, certain derived transition systems are unsatisfactory under this limitation, as Sewell [49] has pointed out. Also we need multi-hole bigraphical contexts, not only to represent parametric reaction rules, but also to admit multiple or 'wide' agents, whose several parts may reside in different regions of a host context.

This gives rise to contexts in which some sites may be active, i.e. may permit reaction to occur, but not others. The following definitions for s-categories allow for this. To do so, we have to introduce a way in which an ordered set of 'places' can be associated with an interface I; we must also indicate, for any arrow $C\colon I \to J$, which places of I 'lie below' a given place in J. This is done by the *width* functor in the following definition; it then leads to the *wide* reactive systems of Definition 3.4.

In what follows we shall use **Ord**, the category of finite ordinals and functions between them.

Definition 3.1 (wide s-category) An s-category $\acute{\mathbf{C}}$ is *wide* if equipped with a functor $\mathsf{width} : \acute{\mathbf{C}} \to \mathbf{Ord}$ with $\mathsf{width}(\epsilon) = 0$ such that, for each bijection π on the ordinal $\mathsf{width}(I)$, there is an isomorphism $\pi_I : I \to I$ in $\acute{\mathbf{C}}$ with $\mathsf{width}(\pi_I) = \pi$.

The objects $I, J, \ldots$ of $\acute{\mathbf{C}}$ are called *interfaces*, and its arrows $A, B, C, \ldots$ are called *contexts*. The domain and codomain of a context will be called its *inner* and *outer faces*. Arrows in a homset $\acute{\mathbf{C}}(\epsilon \to I)$ —which we abbreviate to $\acute{\mathbf{C}}(I)$— are called *ground* arrows; we let lower case letters $a, b, \ldots$ range over these, and abbreviate $a\colon \epsilon \to I$ to $a\colon I$. ∎

We shall later define bigraphs as a wide s-category, since their topography is important. Even in the present general framework we can begin to speak about locality:

Definition 3.2 (location) A *location* of an interface I is a subset $\lambda \subseteq \mathsf{width}(I)$. The width function of a context $C : I \to J$ is extended to locations of I by

$$\mathsf{width}(C)(\lambda) \stackrel{\text{def}}{=} \{\mathsf{width}(C)(i) \mid i \in \lambda\} .$$

The *offset by* n of a location λ is given by $n \dotplus \lambda \stackrel{\text{def}}{=} \{n + i \mid i \in \lambda\}$. ∎

Our definition of reactions for a given Wrs will depend on what it means for a location of I to be active in any context $C\colon I \to J$. The exact meaning will depend on the Wrs, but we need it to satisfy certain natural conditions. This is where the width functor comes in; for example, for $i \in I$ to be active in DC we require both that i is active in C, and that its image under $\mathsf{width}(C)$ is active in D. More precisely:

Definition 3.3 (activity) An *activity* for $\acute{\mathbf{C}}$ is a map $\mathsf{act} : \acute{\mathbf{C}}(I, J) \to 2^{\mathsf{width}(I)}$ for each homset, respecting $\doteqdot$ and satisfying three properties:

(1) $\mathsf{act}(\mathsf{id}_I) = \mathsf{width}(I)$

(2) $i \in \mathsf{act}(DC)$ iff $i \in \mathsf{act}(C)$ and $\mathsf{width}(C)(i) \in \mathsf{act}(D)$

(3) $i \in \mathsf{act}(C \otimes D)$ iff $i \in \mathsf{act}(C)$ or $i - m \in \mathsf{act}(D)$, where $m = \mathsf{width}(\mathsf{dom}(C))$.

For any $C\colon I \to J$ and $\lambda \subseteq \mathsf{act}(C)$ we say that λ is *active in* C, or that C is *active at* λ. If this holds for $\lambda = I$ then we say that C is *active*. ∎

We are now ready to add dynamics to a wide s-category. By enriching it with reaction rules and activity, we shall define a reaction relation over ground arrows.

Definition 3.4 (wide reactive system) A *wide reactive system (Wrs)* $\acute{\mathbf{C}}(\mathsf{act}, \acute{\mathcal{R}})$ is a wide s-category $\acute{\mathbf{C}}$ equipped with an activity act and a set $\acute{\mathcal{R}}$ is a set of *ground reaction rules* of the form $(r\colon I, r'\colon I)$, a *redex* and a *reactum*. Both components must be closed under support translation.

The *reaction relation* $\longrightarrow\!\!\!\triangleright$ over ground arrows is defined as follows: $g \longrightarrow\!\!\!\triangleright g'$ iff there exist a ground reaction rule (r, r') and an active context D with $g \doteqdot Dr$ and $g' \doteqdot Dr'$. ∎

We shall usually denote this Wrs by just $\acute{\mathbf{C}}$. Note that what we have defined are *ground* reaction rules. In a bigraphical reactive system, which is a special kind of Wrs, we shall define a notion of *parametric* reaction rule, each generating a family of ground rules; we have already seen one in Example 2, encoding the dynamics of CCS.

In passing, suppose that we are only concerned with reaction in contexts D that have interfaces of unit width $1 = \{0\}$, so that $\mathsf{width}(D)(0) = 0$. Then D is *active* iff it is active at 0. The first and second activity conditions then amount to requiring that the active contexts form a sub-s-category closed under decomposition. Thus, as promised, we have a proper generalisation of the conditions under which the original congruence theorems [25,26] were proved.

A natural notion of morphism $\mathcal{F}\colon \acute{\mathbf{C}} \to \acute{\mathbf{D}}$ between Wrss is one that preserves width, ground reaction rules and activity. The precise definition is as follows:

Definition 3.5 (Wrs morphism, sub-Wrs) Let $\acute{\mathbf{C}}$ and $\acute{\mathbf{D}}$ be Wrss, and $\mathcal{F}\colon \acute{\mathbf{C}} \to \acute{\mathbf{D}}$ a functor of wide s-categories. Then $\mathcal{F}$ is *morphism* of Wrss if it preserves the ingredients of a Wrs as follows (distinguishing ingredients of $\acute{\mathbf{D}}$ by a prime):

$$\begin{aligned}
\mathsf{width} &= \mathsf{width}' \circ \mathcal{F} \\
(r, r') \in \mathcal{R} &\Rightarrow (\mathcal{F}(r), \mathcal{F}(r')) \in \mathcal{R}' \\
\mathsf{act}(C) &\subseteq \mathsf{act}'(\mathcal{F}(C)) \ .
\end{aligned}$$

If $\mathcal{F}$ is an inclusion functor then we call $\acute{\mathbf{C}}$ a *sub-Wrs* of $\acute{\mathbf{D}}$. ■

Proposition 3.6 (Wrs morphisms preserve reaction) *If $\mathcal{F}\colon \acute{\mathbf{C}} \to \acute{\mathbf{D}}$ is a Wrs morphism, and $g \longrightarrow\!\!\!\triangleright\, g'$ in $\acute{\mathbf{C}}$, then $\mathcal{F}(g) \longrightarrow\!\!\!\triangleright\, \mathcal{F}(g')$ in $\acute{\mathbf{D}}$.*

Clearly Wrss and their morphisms form a category. From now on we shall find it convenient to refer to these morphisms as Wrs functors. An important example of a morphism is the support quotient functor, extended to Wrss as follows:

Definition 3.7 (quotient Wrs) Let $\acute{\mathbf{C}}$ be a Wrs. Then its *support quotient* Wrs is based upon the support quotient $\mathbf{C} = \acute{\mathbf{C}}/\widehat{=}$, with other ingredients as follows:

- the ground reaction rules are $([r], [r'])$, for each rule (r, r') in $\acute{\mathbf{C}}$;
- the active sites of $[D]$ are exactly those of D. ■

Proposition 3.8 (quotient reflects reaction) *The support quotient*

$$[\cdot]\colon \acute{\mathbf{C}} \to \mathbf{C}$$

both preserves and reflects reaction, i.e. $[g] \longrightarrow\!\!\!\triangleright\, [g']$ in $\mathbf{C}$ iff $g \longrightarrow\!\!\!\triangleright\, g'$ in $\acute{\mathbf{C}}$.

The quotient morphism takes a *concrete* Wrs, based on an s-category, to an *abstract* Wrs based upon a category. In the next section we show how to obtain a behavioural congruence for an arbitrary concrete Wrs $\acute{\mathbf{C}}$ with sufficient RPOs. The support quotient $\mathbf{C}$ of $\acute{\mathbf{C}}$ may not possess RPOs, but nevertheless the quotient functor allows us to derive a behavioural congruence for $\mathbf{C}$ also. This use of a concrete Wrs with RPOs to supply a behavioural congruence for the corresponding abstract Wrs was first represented by the *functorial reactive systems* of Leifer's thesis [25].

4 Transition systems

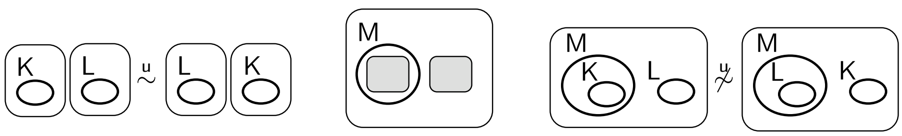
We now consider how to equip a Wrs with a labelled transition system. This will comprise a set of ground arrows called *agents*, together with a set of transitions of a form such as $a \xrightarrow{L} a'$, where a, a' are agents and L is a context with La defined. Then bisimilarity is defined in the usual way, and we are interested in general conditions under which it will be a congruence.

Leifer and Milner [26] defined labelled transitions as follows: $a \xrightarrow{L} a'$ if there is a reaction rule (r, r') and an active context D for which (L, D) is an idem pushout (IPO) for (a, r) and $a' = Dr'$. We shall adopt a slight refinement of this definition; we shall equip a transition with information about locality. For an agent $a\colon I$, a transition of the form $a \xrightarrow{L} a'$ tells us the extra context $L\colon I \to J$ needed by a to create a redex, but does not specify *where* this completed redex occurs within La, i.e. within which region(s) the reaction takes place. Such regions are identified by a location λ of J, the outer face of L. It turns out that, to achieve congruence of bisimilarity, we must index each transition by such a location. This can be illustrated by a simple example, for which we need only the superficial understanding of bigraphs supplied by the Introduction.

Example 3 (non-congruence) This example shows that bisimilarity based upon unlocated transitions, which we denote by $\overset{\mathsf{u}}{\sim}$, is not in general a congruence for bigraphical systems. Take controls K, L and M, with M passive. Links are irrelevant in this example, so we take interfaces to be just finite ordinals (widths).

Now write $\mathsf{K}, \mathsf{L} : 0 \to 1$ for *atoms*, i.e. a single node with no content, and $\mathsf{M} : 1 \to 1$ for the passive context consisting of a single M-node. Let there be a single reaction rule (K, L); it allows the reaction $\mathsf{K} \longrightarrow \mathsf{L}$ in any active context.

Consider the two agents $a, b : 0 \to 2$ illustrated below, where $a = \mathsf{K} \otimes \mathsf{L}$ and $b = \mathsf{L} \otimes \mathsf{K}$. They can each do a transition that turns K into L, i.e. we have $a \xrightarrow{\mathsf{id}_2} \mathsf{L} \otimes \mathsf{L}$ and $b \xrightarrow{\mathsf{id}_2} \mathsf{L} \otimes \mathsf{L}$. Because these two transitions do not record the different places at which the reaction occurs, it turns out that $a \overset{\mathsf{u}}{\sim} b$.



$a \overset{\mathsf{u}}{\sim} b$ $\qquad$ $C \overset{\text{def}}{=} \mathsf{M} \mid \mathsf{id}_1$ $\qquad$ $Ca \overset{\mathsf{u}}{\not\sim} Cb$

Now putting a and b in the context $C \overset{\text{def}}{=} \mathsf{M} \mid \mathsf{id}_1 \colon 2 \to 1$, we find $Ca \overset{\mathsf{u}}{\not\sim} Cb$. In Cb the K-node can react, so there is a transition $Cb \xrightarrow{\mathsf{id}_1}$; but Ca has no such transition since its K-node cannot react. ■

We are now ready to define transition systems. We allow for a broad class of transitions, within which we distinguish those based upon IPOs.

$$\begin{tikzcd}
\cdot \arrow[r, "L"] & \cdot & \\
\cdot \arrow[u, "a"] \arrow[r, "r"'] & \cdot \arrow[u, "D"'] & \\
& & \cdot \arrow[ul, "r'"'] \arrow[uul, bend right, "\bumpeq a'"']
\end{tikzcd}$$

Definition 4.1 (transition) A *transition* consists of a quadruple (a, L, λ, a'), written $a \xrightarrow{L}_\lambda a'$, with a and a' ground, such that $La = Dr$ and $a' \bumpeq Dr'$ for some ground reaction rule $(r, r' \colon I)$ and active D such that $\lambda = \mathsf{width}(D)(\mathsf{width}(I))$.

We say that the reaction rule and the diagram $La = Dr$ *underlie* the transition. A transition is *minimal* if the underlying diagram is an IPO. ■

Definition 4.2 (transition system) Given a Wrs $\acute{\mathbf{C}}$, a *(labelled) transition system* for $\acute{\mathbf{C}}$ is a pair $\mathcal{L} = (\mathcal{A}, \mathcal{T})$, where $\mathcal{A}$ is a set of ground arrows, the *agents* of $\mathcal{L}$, and $\mathcal{T}$ is a set of transitions $a \xrightarrow{L}_\lambda a'$ with $a, a' \in \mathcal{A}$.

We abbreviate '(labelled) transition system' to Lts. An Lts $\mathcal{M}$ is a *sub-Lts* of $\mathcal{L}$, written $\mathcal{M} \prec \mathcal{L}$, if its components are included respectively in those of $\mathcal{L}$. ■

The *full* (resp. *standard*) transition system for a Wrs consists of all interfaces, together with all (resp. all minimal) transitions. When the Wrs concerned is understood we shall denote these two transition systems respectively by FT and ST.

Returning briefly to Example 3 we now see that the location component in transitions allows us to distinguish between the two agents a and b. In fact in ST their only transitions are $a \xrightarrow{\mathsf{id}}_{\{0\}} \mathsf{L} \otimes \mathsf{L}$ and $b \xrightarrow{\mathsf{id}}_{\{1\}} \mathsf{L} \otimes \mathsf{L}$.

Our concept of Lts is very broad, and we shall be interested in those that behave well in various ways. For example, if two agents are support equivalent, then we expect their transitions to 'agree'. We define what this means more generally, for an arbitrary congruence relation.

Definition 4.3 (respect) Let $\equiv$ be a congruence in a Wrs equipped with $\mathcal{L}$. Then $\equiv$ and $\mathcal{L}$ are said to *respect* one another if the following holds:

Let $a \xrightarrow{L}_\lambda a'$ be a transition in $\mathcal{L}$. Let $a \equiv b$ and $L \equiv M$, with Mb defined. Then there exist an agent b' and a transition $b \xrightarrow{M}_\lambda b'$ in $\mathcal{L}$ such that $a' \equiv b'$. ■

'Respect' is mutual between an equivalence and an Lts, so that '$\mathcal{L}$ respects $\equiv$' means the same as '$\equiv$ respects $\mathcal{L}$'; we shall use them interchangeably.

Our definition of transition presupposes a set of reaction rules, i.e. an *unlabelled* transition relation. Sometimes, for example in CCS, labelled transition systems have been the primary means of providing process dynamics, and unlabelled transitions corresponded to transitions with a 'null' label (τ in CCS). But in this work we are committed to taking reaction rules as primary, because they can be described generally without presupposing any notion of a transition label.

Whether transitions are derived from reactions or defined in some other way, we may use them to define behavioural equivalences and preorders. Here we shall limit attention to strong bisimilarity. (Throughout this paper we shall omit 'strong' since we do not define or use weak bisimilarity.)

Definition 4.4 (wide bisimilarity) Let $\acute{\mathbf{C}}$ be a wide reactive system equipped with an Lts $\mathcal{L}$. A *simulation (on $\mathcal{L}$)* is a binary relation $\mathcal{S}$ between agents with equal interface such that if $a\mathcal{S}b$ and $a \xrightarrow{L}_{\lambda} a'$ in $\mathcal{L}$, then whenever Lb is defined there exists b' such that $b \xrightarrow{L}_{\lambda} b'$ in $\mathcal{L}$ and $a'\mathcal{S}b'$. A *bisimulation* is a symmetric simulation. Then *bisimilarity (on $\mathcal{L}$)*, denoted by $\sim_{\mathcal{L}}$, is the largest bisimulation (on $\mathcal{L}$). ∎

We shall often omit 'on $\mathcal{L}$', and write $\sim$ for $\sim_{\mathcal{L}}$, when $\mathcal{L}$ is understood from the context. This will usually be when $\mathcal{L}$ is ST.

Note the slight departure from the usual definition of bisimulation of Park [41]; here we require Lb to be defined. This is merely a technical detail, provided that the Lts respects support equivalence; for then, whenever La is defined there will always exist $L' \simeq L$ for which *both* $L'a$ and $L'b$ are defined. Moreover if the Wrs is based on a category, in particular a support quotient, then the side-condition holds automatically; in that case the definition of bisimilarity reduces to the standard one.

If $\mathcal{S}$ is a binary relation and $\equiv$ an equivalence, then we define $\mathcal{S}^{\equiv}$ to be the closure of $\mathcal{S}$ under $\equiv$, i.e. the relational composition $\equiv\mathcal{S}\equiv$. It is well known [30] that if $\equiv$ is included in (strong) bisimilarity, then to establish bisimilarity it is enough exhibit a *bisimulation up to* $\equiv$; that is, a symmetric relation $\mathcal{S}$ such that whenever $a\mathcal{S}b$ then each transition of a is matched by b in $\mathcal{S}^{\equiv}$. We now deduce from Proposition 2.10(5) that support equivalence can be used in this way:

Proposition 4.5 (support translation of transitions) *In a Wrs the full and standard transition systems respect support equivalence. Hence in each case $\simeq$ is a bisimulation, and a bisimulation up to $\simeq$ suffices to establish bisimilarity.*

We now come to our congruence theorem for a Wrs; the proof is in [27].

Theorem 4.6 (congruence of wide bisimilarity) *In a Wrs with RPOs, equipped with the standard transition system, wide bisimilarity of agents is a congruence; that is, if $a_0 \sim a_1$ then $Ca_0 \sim Ca_1$.*

We shall henceforth often omit the adjective 'wide' when discussing bisimilarity. Recall that we are taking (strong) bisimilarity as a representative of many preorders and equivalences; Leifer [25] has proved congruence theorems for several others, and we expect that those results can be transferred to the present setting. Furthermore Jensen [21] extends our theory smoothly to weak transitions and weak bisimilarity.

Since there are many transition systems, there are also many variants of bisimilarity. Some are congruences, some are not. For example, the above proof is easily adapted to show the congruence of full bisimilarity, which is based upon *all* transitions, not just those based on IPOs. But we have already commented on the unsatisfactory nature of FT, and the fact that it involves an intractable family of labels.

In Section 11 we shall identify a transition system whose bisimilarity is a congruence for a limited range of contexts, and exactly matches the original bisimilarity of finite CCS (Milner, [31]).

Quotient transition systems

Let us now turn to transition systems derived for a quotient Wrs.

Definition 4.7 (transitions for a quotient Wrs) Let $\acute{\mathbf{C}}$ be a Wrs equipped with an Lts $\mathcal{L} = (\mathcal{A}, \mathcal{T})$, and let $\mathcal{F} : \acute{\mathbf{C}} \to \acute{\mathbf{D}}$ be a Wrs functor. We say that $\mathcal{F}$ *respects* $\mathcal{L}$ if the congruence it induces on $\acute{\mathbf{C}}$ respects $\mathcal{L}$.

The Lts $\mathcal{F}(\mathcal{L})$ *induced* by $\mathcal{F}$ on $\acute{\mathbf{D}}$ has agents $\mathcal{F}(\mathcal{A}) \stackrel{\text{def}}{=} \{\mathcal{F}(a) \mid a \in \mathcal{A}\}$. Whenever $\mathcal{L}$ has a transition $a \xrightarrow{L}\!\!\triangleright_{\lambda} a'$ then $\mathcal{F}(\mathcal{L})$ has the transition

$$\mathcal{F}(a) \xrightarrow{\mathcal{F}(L)}\!\!\triangleright_{\lambda} \mathcal{F}(a') \ . \qquad \blacksquare$$

This definition always makes sense, but it will not always make bisimilarity a congruence in $\acute{\mathbf{D}}$, even if it is so in $\acute{\mathbf{C}}$. However the next theorem, proved in Appendix A, tells us when this will be ensured. Recall that a *full* functor is surjective for each homset.

Theorem 4.8 (transitions induced by functors) *Let $\acute{\mathbf{C}}$ be equipped with an Lts $\mathcal{L}$. Let $\mathcal{F} : \acute{\mathbf{C}} \to \acute{\mathbf{D}}$ be a full Wrs functor that is the identity on objects and respects $\mathcal{L}$, and such that $\mathcal{F}(a) = \mathcal{F}(b)$ whenever $a \bumpeq b$. Then the following hold for $\mathcal{F}(\mathcal{L})$:*

(1) $a \sim b$ in $\acute{\mathbf{C}}$ iff $\mathcal{F}(a) \sim \mathcal{F}(b)$ in $\acute{\mathbf{D}}$.
(2) If bisimilarity is a congruence in $\acute{\mathbf{C}}$ then it is a congruence in $\acute{\mathbf{D}}$.

These results prepare the way for setting up a bigraphical reactive system (Brs) as a Wrs, and then deriving an Lts and behavioural congruences for it. We typically want to do this for an *abstract* Brs, i.e. one based upon a category where support equivalence has been factored out, rather than for a *concrete* Brs based upon an s-category, where arrows (bigraphs) have non-trivial support. For example, CCS and Petri nets are naturally formulated as abstract Brss. But the RPOs needed to derive satisfactory Ltss are typically not present in abstract Brss; this is implied by a counter-example, Example 6 in Section 6.

Now, as we shall see in Section 9, a Brs (concrete or abstract) is determined by a signature $\mathcal{K}$ and a set $\mathcal{R}$ of reaction rules. So our procedure for establishing an Lts and congruential bisimilarity for an abstract Brs will be as follows:

- Define an abstract Brs $\mathbf{C}(\mathcal{K},\mathcal{R})$;
- Define a concrete Brs $\acute{\mathbf{C}}(\mathcal{K},\acute{\mathcal{R}})$, of which $\mathbf{C}(\mathcal{K},\mathcal{R})$ is the quotient (and $\mathcal{R}$ the quotient of $\acute{\mathcal{R}}$) by some equivalence $\equiv$;
- Derive an Lts for $\acute{\mathbf{C}}(\mathcal{K},\acute{\mathcal{R}})$ with an associated behavioural congruence, and ensure that it respects $\equiv$;
- Use Definition 4.7 to transfer the Lts to $\mathbf{C}(\mathcal{K},\mathcal{R})$, and Theorem 4.8 to ensure a behavioural congruence in the abstract Brs.

This procedure is illustrated for finite CCS in Section 11.

Adequate and definite transition systems

We now turn to a question that arises strongly in applications. Our standard Lts, containing only the minimal transitions, is of course much smaller than the full Lts. But it turns out that in particular cases we can reduce the standard Lts still further, without affecting bisimilarity. We introduce here the basic concepts to make this idea precise, since they do not depend at all on the notion of bigraph.

Definition 4.9 (relative bisimulation, adequacy) Let $\mathcal{M} \prec \mathcal{L}$. A *relative bisimulation for $\mathcal{M}$ (on $\mathcal{L}$)* is a symmetric relation $\mathcal{S}$ such that

> whenever $a\mathcal{S}b$, then for every transition $a \xrightarrow{L}\!\!\triangleright_{\lambda} a'$ in $\mathcal{M}$, with Lb defined, there exists b' such that $b \xrightarrow{L}\!\!\triangleright_{\lambda} b'$ in $\mathcal{L}$ and $a'\mathcal{S}b'$.

Then *relative bisimilarity for $\mathcal{M}$ on $\mathcal{L}$*, denoted by $\sim^{\mathcal{M}}_{\mathcal{L}}$, is the largest relative bisimulation for $\mathcal{M}$ on $\mathcal{L}$. We call $\mathcal{M}$ *adequate for $\mathcal{L}$* if $\sim^{\mathcal{M}}_{\mathcal{L}}$ coincides with $\sim_{\mathcal{L}}$ on the agents of $\mathcal{M}$; if $\mathcal{M}$ has agents $\mathcal{A}$, we write this as $\sim^{\mathcal{M}}_{\mathcal{L}} = \sim_{\mathcal{L}} \restriction \mathcal{A}$. ■

Note that, for $a \sim^{\mathcal{M}}_{\mathcal{L}} b$, we require b only to match the transitions of a that lie in $\mathcal{M}$, and b's matching transition need not lie in $\mathcal{M}$. This means that relative bisimilarity is in general not transitive, so it is not in itself a behavioural equivalence.

Relative bisimilarity is useful when $\mathcal{M}$ is adequate for $\mathcal{L}$; it reduces the class of transitions to be checked. For example, usually fewer labels are involved.

In the case that $\mathcal{L}$ is ST we can give a simple example of adequacy. It depends upon the fact that ST is *closed under isomorphism*, i.e. if $a \xrightarrow{L}_\lambda a'$ is a transition of ST then so is $\iota a \xrightarrow{\kappa L \iota^{-1}}_\lambda \kappa a'$ for any isos ι and κ. Then when checking for bisimilarity with a given a, intuitively it should suffice to consider not *every* transition of a, but only one in every iso class. Thus these representative transitions should constitute an adequate Lts. In fact this is true more generally (for a proof see [26]):

Proposition 4.10 (representative transitions) *Let $\mathcal{L}$ be an Lts closed under isomorphism, and let $\mathcal{M} \prec \mathcal{L}$. Suppose that, for every transition $a \xrightarrow{L}_\lambda a'$ in $\mathcal{L}$, there is a transition $a \xrightarrow{\kappa L}_\lambda \kappa a'$ in $\mathcal{M}$ for some iso κ. Then $\mathcal{M}$ is adequate for $\mathcal{L}$.*

A deeper example of adequacy arises when we consider *parametric* reaction rules; such a rule has a *parametric* redex R, and generates a family of ground rules whose redexes take the form $r = Rd$ where d is a parameter. Most interesting reaction rules, e.g. in the λ-calculus, take this form; indeed we shall adopt it in bigraphical reactive systems, as already illustrated for CCS in Section 1 (Figure 2). Our intuition is that the important transitions are those where the agent contributes significantly to the underlying parametric redex. We can make this precise in terms of support: we are interested in transitions of a whose underlying parametric redex R is such that $|a| \cap |R| \neq \emptyset$. We call such transitions *engaged*. We may naturally expect that the engaged transitions are adequate. In Section 9 we shall later prove this for a particular class of bigraphical reactive systems, the *simple* ones. In Section 11 we shall see in the case of CCS that this greatly simplifies behavioural analysis.

We now look at a well-behaved kind of sub-Lts. For arbitrary $\mathcal{M} \prec \mathcal{L}$ and any given pair (L, λ), it is possible that $\mathcal{M}$ contains some but not all of the (L, λ)-transitions in $\mathcal{L}$. If this is not the case then the situation is somewhat simpler.

Definition 4.11 (definite sub-Lts) Let $\mathcal{M} \prec \mathcal{L}$. Call $\mathcal{M}$ *definite for* $\mathcal{L}$ if, for any transition $a \xrightarrow{L}_\lambda a'$ of $\mathcal{L}$, the pair (L, λ) alone determines whether it lies in $\mathcal{M}$.

In this case we find that a relative bisimilarity is an absolute one:

Proposition 4.12 (definite sub-Lts) *If $\mathcal{M}$ is definite for $\mathcal{L}$ then $\sim_\mathcal{M} = \sim_\mathcal{L}^\mathcal{M}$.*

An important consequence is that, if we know bisimilarity to be a congruence on $\mathcal{L}$, then the same holds for any $\mathcal{M}$ definite and adequate for $\mathcal{L}$. In fact:

Corollary 4.13 (adequate congruence) *Let $\mathcal{M}$ be definite and adequate for $\mathcal{L}$. Then*

(1) The bisimilarities on $\mathcal{M}$ and $\mathcal{L}$ coincide on the $\mathcal{A}$, the agents of $\mathcal{M}$.
(2) Let C be a context that preserves $\sim_\mathcal{L}$, and also preserves membership of $\mathcal{A}$. Then C preserves $\sim_\mathcal{M}$.

Part II : Bigraphical structure

Section 5 defines the notion of a *concrete pure bigraph* formally, in terms of its two constituents: a *place graph* representing locality and a *link graph* representing connectivity. Sections 6 and 7 define these two notions in turn, ensuring that they enjoy the necessary categorical properties, including RPOs. Section 8 then combines these constituents, yielding a theory of pure bigraphs where locality and connectivity are independent. It defines important properties and operations for bigraphs; it also introduces a quotient functor from concrete to *abstract* bigraphs, where support is forgotten and the notions of occurrence and RPO are lost.

5 Pure bigraphs: definition

In this section we define the notion of *pure bigraph* formally, in terms of the constituent notions of *place graph* and *link graph*, which are dealt with in the following two sections. Let us begin with an illustration.

Example 4 (resolving a bigraph) An example of a bigraph appeared in Figure 1; it illustrated how nodes are nested, and how —independently of the nesting— they are linked via their ports. Figure 3 shows another example, illustrating more of the

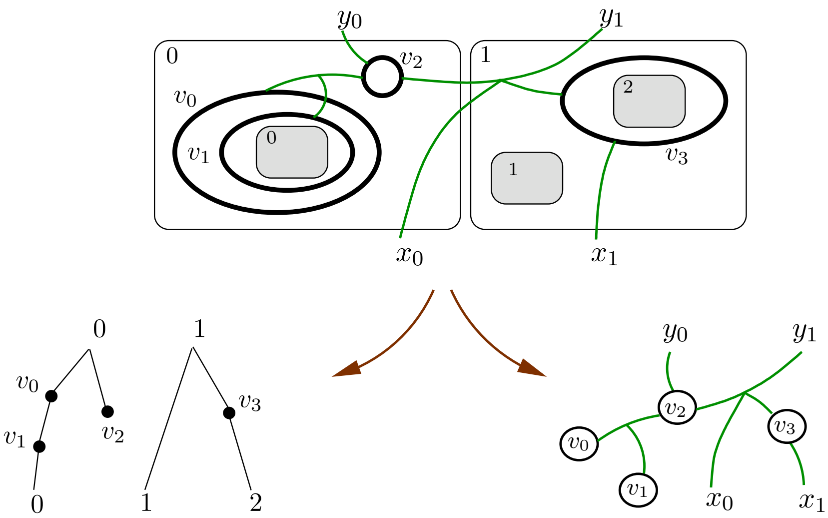


Fig. 3. Resolving a bigraph into a place graph and a link graph

structure of bigraphs. First, it shows how a bigraph may be resolved into its two constituents, a *place graph* and a *link graph*. This is what we mean by the independence of placing and linking; the place graph (a forest) is completely independent of the link graph (a kind of hypergraph) as long as they shared the same node set, here $\{v_0, \ldots, v_3\}$. (Controls are not shown in this example.) If we forget everything in the bigraph except the nesting of regions (large squares), nodes and sites (grey holes) then we get the place graph; if on the other hand we forget this nesting but retain the linkage, we get the link graph. From our definitions it will be clear that these two projections are full functors.

Using this example we can also describe composition. Our bigraph has width 2 (two regions), so it can inserted in a host graph having two sites. It also has *outer names* y_0, y_1; this means that the host bigraph must have these *inner names*, allowing linkage to be formed by composition. Equally, our bigraph has three sites (grey holes) and inner names $\{x_0, x_1, x_2\}$; these provide for composition with a three-region client that possesses these outer names. Then composition of two bigraphs can be described thus: first resolve into constituents, then compose these, and finally combine two larger constituents into a bigraph. ■

We are now ready for a formal definition.

Definition 5.1 (pure signature) A *(pure) signature* $\mathcal{K}$ is a set whose elements are called *controls*. For each control K it provides a finite ordinal $ar(K)$, an *arity*; it also determines which controls are *atomic*, and which of the non-atomic controls are *active*. Controls which are not active (including the atomic controls) are called *passive*. ■

Note that a signature need not be finite, or even denumerable. Thus a bigraph, though itself finite, may denote an element of a continuous state space.

As we saw in Example 2 in Section 1, a non-atomic node —one with a non-atomic control— may contain other nodes. A node's control determines its ports, and if the control is active then reactions are permitted inside the node. A passive node — such as a get-node in the CCS example— can be thought of as a script, or program, awaiting activation; this must take the form of a reaction that destroys the node boundary.

In refinements of the theory a signature may carry further information, such as a *sign* and/or a *sort* for each port. The sign may be used, for example, to enforce the restriction that each negative port is connected to exactly one positive port, as in action calculi [10,32]. Sorting of ports has been used to model Petri nets as bigraphs [36,27]. Another possible refinement is to assign a sort to each control K, determining the possible controls for the children of any K-node; we illustrate this in modelling CCS in Section 11. In [23] we also defined an important refinement that allows names to have *scope*, and controls to *bind* names. The theory of *pure* bigraphs is prerequisite to understanding all these refinements.

We presuppose a denumerable set $\mathcal{X}$ of *names*. We shall define *concrete* bigraphs top-down; here we define a bigraph as the combination of two constituents, and in the following sections we define those constituents themselves.

Definition 5.2 (concrete pure bigraph) A *(concrete) pure bigraph* over the signature $\mathcal{K}$ takes the form $G = (V, E, ctrl, G^{\mathsf{P}}, G^{\mathsf{L}}) : I \to J$ where $I = \langle m, X \rangle$ and $J = \langle n, Y \rangle$ are its *inner* and *outer faces*, each combining a *width* (a finite ordinal) with a finite set of global names drawn from $\mathcal{X}$. Its first two components V and E are finite sets of *nodes* and *edges* respectively. The third component $ctrl \colon V \to \mathcal{K}$, a *control map*, assigns a control to each node. The remaining two are:

$$G^{\mathsf{P}} = (V, ctrl, prnt) \colon m \to n \qquad \text{a } \textit{place graph}$$

$$G^{\mathsf{L}} = (V, E, ctrl, link) \colon X \to Y \ \text{ a } \textit{link graph} \, .$$

Place graphs and link graphs are defined in Definitions 6.1 and 7.1 respectively. We call G the *combination* of its *constituents* G^{P} and G^{L}, writing $G = \langle G^{\mathsf{P}}, G^{\mathsf{L}} \rangle$. ■

In *concrete* bigraphs the nodes and edges have identity. The support of a concrete bigraph consists of its nodes and edges; in terms of the definition, $|G| = V + E$. We shall work with s-categories of bigraphs, because RPOs exist there.

In Section 8 we revisit bigraphs in order to develop their structure, often by combining attributes of their constituent place graphs and link graphs. In that section we shall also take the quotient by support equivalence to obtain *abstract* bigraphs. Until then, unless otherwise stated we shall be concerned with concrete bigraphs, place graphs and link graphs so we shall omit 'concrete'.

6 Place graphs

Definition 6.1 (place graph) A *place graph* $A = (V, ctrl, prnt) : m \to n$ has an *inner width* m and an *outer width* n, both finite ordinals; a finite set V of nodes with a control map $ctrl : V \to \mathcal{K}$; and a *parent map* $prnt : m \uplus V \to V \uplus n$. We write $w >_A w'$, or just $w > w'$, to mean $w = prnt^k(w')$ for some $k > 0$. The parent map is *acyclic*, i.e. we insist that $>_A$ is a partial order. An *atom*, i.e. a node with atomic control, cannot be a parent.

The widths m and n index the *sites* and *roots* of A respectively. The sites and nodes —i.e. the domain of $prnt$— are called *places*. A place graph is *hard* if every root, and every node except an atom, has a child. ■

In this paper we shall limit attention to hard place graphs. We shall therefore omit the adjective 'hard'; but we retain a subscript h in the name of the s-category as a reminder. Although most of the work can be done without this limitation, it simplifies certain aspects. For example, hard place graphs possess pushouts; also, Theorem 9.11 (adequacy) appears to need an extra condition if we admit arbitrary place graphs.

Due to acyclicity, a place graph with outer width n is an ordered sequence of n unordered trees. The sites and roots provide the means of composing two place graphs; each root of the first is planted in the corresponding site of the second. Figure 4 shows two simple examples of composition, B_0A_0 and B_1A_1. Formally:

Definition 6.2 (s-category of place graphs) The s-category $\acute{\mathrm{P}}\mathrm{LG}_\mathsf{h}$ has finite ordinals as objects and (hard) place graphs as arrows. The support of a place graph is its node set. The composition $A_1A_0 : m_0 \to m_2$ of two place graphs

$$A_i = (V_i, ctrl_i, prnt_i) : m_i \to m_{i+1} \ (i = 0, 1)$$

with disjoint supports is $A_1A_0 \stackrel{\text{def}}{=} (V, ctrl, prnt)$, where $V = V_0 \uplus V_1$, $ctrl = ctrl_0 \uplus ctrl_1$, and $prnt = (\mathsf{Id}_{V_0} \uplus prnt_1) \circ (prnt_0 \uplus \mathsf{Id}_{V_1})$. The identity place graph at m is $\mathsf{id}_m \stackrel{\text{def}}{=} (\emptyset, \emptyset_{\mathcal{K}}, \mathsf{Id}_m) : m \to m$.

The *tensor product* $\otimes$ in $\acute{\mathrm{P}}\mathrm{LG}_\mathsf{h}$ is defined as follows: On objects, we take $m \otimes n \stackrel{\text{def}}{=} m + n$. For the product $A_0 \otimes A_1$ of two place graphs with disjoint support we take the union of their node sets; for the parent map, if $A_0 \colon m_0 \to n_0$, we first offset the sites and roots of A_1 by m_0 and n_0 respectively, then take the union of the two parent maps.

For an injective map ρ on nodes, the support translation $\rho \cdot A$ is defined by systematic replacement of each node v by $\rho(v)$, preserving all structure. ■

It is easy to check that the equations for an s-category are satisfied.

In Section 2 we motivated the notion of support set of an arrow in an s-category, by claiming that it would serve to distinguish between occurrences of the same entity within a larger entity. Note that a node in a place graph has an identity (its node name) as well as a control that tells us what kind of node it is. If we do not know the identity of a node, then we cannot distinguish between two nodes with the same control. It follows, as we shall see shortly, that if we forget the identity of nodes we arrive at a category of 'abstract' place graphs in which RPOs do not exist. This means that our method of deriving transition systems becomes inapplicable for abstract place graphs. We justify this remark at the end of the section; meanwhile we shall develop the theory of hard place graphs far enough to discover they possess not only RPOs, but even pushouts.

Definition 6.3 (sibling, active, passive) Two places are *siblings* if they have the same parent. A site s of A is *active* if $ctrl(v)$ is active whenever $v >_A s$; otherwise s is *passive*. If s is active (resp. passive) in A, we also say that A is *active* (resp. *passive*) *at* s. ■

When dealing with many place graphs A, B, ... we may index their parent maps as $prnt_A$, $prnt_B$ etc. At times we shall find it more convenient to abuse notation and denote the parent map of a place graph A again by A. The context will prevent ambiguity; for example in BA we are talking of place graphs, while in $B(A(v))$ we are talking of their parent maps. Thus $(BA)(v)$ means the parent map of the composite place graph BA applied to the node v.

Proposition 6.4 (isomorphisms in place graphs) *An arrow $\iota : m \to m$ in* $\grave{\mathrm{P}}\mathrm{LG}_\mathsf{h}$ *is an isomorphism iff it has no nodes, and its parent map is a bijection.*

Epimorphisms (epis) will play a central role, both for place graphs and for link graphs. Recall that an arrow A is epi if $BA = CA$ implies $B = C$. Monomorphisms (monos), the dual of epis, will also be used. Let us call a place graph *inner-injective* if no two sites are siblings (i.e. the parent map restricted to sites is injective). Then:

Proposition 6.5 (epis and monos in place graphs) *In* $\grave{\mathrm{P}}\mathrm{LG}_\mathsf{h}$*, every place graph is epi; a place graph is mono iff it is inner-injective.*

This is analogous to the category of sets with functions, where the epis and monos are the surjective and injective functions respectively. Indeed, in hard place graphs the parent map is always surjective on roots; and to say that no two sites are siblings is just to say that the parent map is injective from sites.

A related fact is that not only RPOs but pushouts exist in $\acute{\mathrm{P}}\mathrm{LG}_\mathsf{h}$, but only for pairs $\vec{A}\colon h \to \vec{m}$ that possess a bound. Before giving the construction of pushouts, we give three conditions on $\vec{A}$ that will turn out to be necessary and sufficient for a bound, and furthermore for a pushout. Roughly speaking, the conditions ensure that A_0 and A_1 treat their shared sites and nodes compatibly; then a bound $\vec{B}$ can exist, since B_0 can extend A_0 to include 'the part of A_1 not shared with A_0'. Such a bound will also be a pushout if, roughly, it adds no more than necessary for this.

Notation When considering a pair $\vec{A} : h \to \vec{m}$ of place graphs with common sites h, we shall adopt a convention for naming their nodes. We denote the node set of A_i ($i = 0, 1$) by V_i, and denote $V_0 \cap V_1$ by V_2. Recall that $\bar{\imath}$ means $1-i$ for $i \in 2$. We shall use $v_i, v_i', \ldots$ to range over V_i ($i = 0, 1, 2$), and r_i, r_i' to range over the roots m_i ($i = 0, 1$). We shall also use $w_2, w_2', \ldots$ to range over $h \uplus V_2$; this is useful because shared sites behave just like shared nodes in our construction of pushouts. ■

Definition 6.6 (consistency conditions for place graphs) We define three *consistency* conditions on a pair $\vec{A} : h \to \vec{m}$ of place graphs.

CP0 $\mathit{ctrl}_0(v_2) = \mathit{ctrl}_1(v_2)$

CP1 If $A_i(w) \in V_2$ then $w \in h \uplus V_2$ and $A_{\bar{\imath}}(w) = A_i(w)$

CP2 If $A_i(w_2) \in V_i \setminus V_2$ then $A_{\bar{\imath}}(w_2) \in m_{\bar{\imath}}$, and if also $A_{\bar{\imath}}(w) = A_{\bar{\imath}}(w_2)$ then $w \in h \uplus V_2$ and $A_i(w) = A_i(w_2)$. ■

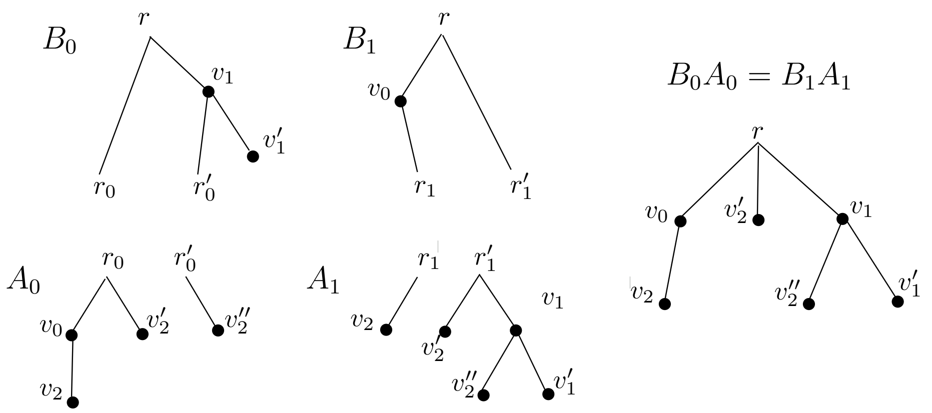


Fig. 4. A consistent pair $\vec{A}$ of place graphs, with bound $\vec{B}$

Let us express CP1 and CP2 in words; they are both to do with children of nodes. If $i = 0$, CP1 says that if the parent of a place w in A_0 is a node shared with A_1, then w is also shared and has the same parent in A_1. CP2 says, on the other hand, that if the parent of a shared place w_2 in A_0 is an *unshared* node, then its parent in A_1 must be a root, and any sibling of w_2 in A_1 must also be its sibling in A_0.

Necessity of these conditions is easy, and we omit the proof:

Proposition 6.7 (consistency in place graphs) *If the pair $\vec{A}$ has a bound, then the consistency conditions hold.*

Before going further, it may be helpful to see a simple example.

Example 5 (consistent place graphs) Consider the pair $\vec{A}$ in Figure 4, each with two roots and no sites; nodes with subscript 2 are shared. (Controls are not shown). It is worth checking that the consistency conditions hold, and that indeed $\vec{B}$ is a bound.

What happens if a node u is added to A_1 as a sibling of v_2? If u is unshared then CP2 is violated, so no bound exists. If u is shared, then to preserve the consistency conditions it must also be a sibling of v_2 in A_0; then $\vec{B}$ remains a bound. ■

Now, assuming the consistency conditions of Definition 6.6, we shall prove that there exists a pushout for $\vec{A}$. (Thus, since any pushout is a bound, we shall also have shown that the consistency conditions are sufficient for a bound to exist.)

Construction 6.8 (pushouts in place graphs) Assume the consistency conditions for the pair of place graphs $\vec{A} : h \to \vec{m}$. We define a pushout $\vec{C} : \vec{m} \to n$ for $\vec{A}$ as follows.

nodes: Take the nodes of C_i to be $V_{\bar{i}} \setminus V_2$.

interface: Define $m'_i \subseteq m_i$, the roots to be mapped to the codomain n, by

$$m'_i \stackrel{\text{def}}{=} \{r \in m_i \mid \forall w_2 \in h \uplus V_2.\ A_i(w_2) = r \Rightarrow A_{\bar{\imath}}(w_2) \in m_{\bar{\imath}}\} \ .$$

Next, on the disjoint sum $m'_0 + m'_1$, define $\simeq$ to be the smallest equivalence such that $(0, r_0) \simeq (1, r_1)$ whenever $A_0(w_2) = r_0$ and $A_1(w_2) = r_1$ for some w_2. Then define the codomain up to isomorphism by

$$n \stackrel{\text{def}}{=} (m'_0 + m'_1)/\!\simeq \ .$$

For each $r \in m'_i$ we denote the $\simeq$-equivalence class of (i, r) by $\widehat{i, r}$.

parents: Define the parent map of $C_0 : m_0 \to n$ as follows (C_1 is similar):

For $r \in m_0$:

$$C_0(r) \stackrel{\text{def}}{=} \begin{cases} \widehat{0, r} & \text{if } r \in m'_0 \\ A_1(w_2) & \text{if } r \notin m'_0, \text{ for some } w_2 \text{ with } A_0(w_2) = r \end{cases}$$

For $v \in V_1 \setminus V_2$:

$$C_0(v) \stackrel{\text{def}}{=} \begin{cases} \widehat{1, r} & \text{if } A_1(v) = r \in m_1 \\ A_1(v) & \text{if } A_1(v) \notin m_1 \ . \end{cases}$$ ∎

It is straightforward to check that each C_i is hard. We also have to show that the definition is sound. Thus in the second clause for $C_0(r)$ we must ensure that $w_2 \in h \uplus V_2$ exists such that $A_0(w_2) = r$, and that each such w_2 yields the same value $A_1(w_2)$ in $V_1 \setminus V_2$; in the first clause for $C_0(v)$ we must ensure that $r \in m'_1$. The consistency conditions do ensure this, and also that $C_0 A_0 = C_1 A_1$.

We now validate our construction:

Theorem 6.9 (valid pushout construction) *If the pair $\vec{A} : h \to \vec{m}$ is consistent then the pair $\vec{C} : \vec{m} \to n$ defined by Construction 6.8 is a pushout for $\vec{A}$.*

Proof (outline) Let $\vec{B}$ be any bound for $\vec{A}$. We define a mediating arrow D such that $DC_i = B_i$ $(i = 0, 1)$ as follows. The nodes of D are those in $\vec{B}$ not in $V_0 \cup V_1$, and for any such node v define $D(v) \stackrel{\text{def}}{=} B_i(v)$ $(i = 0, 1)$. It remains to define $D(s)$ for $s \in n$. We have $s = \widehat{i, r}$ for $r \in m'_i$, for $i = 0$ or $i = 1$ or both. In either case, set $D(s) \stackrel{\text{def}}{=} B_i(r)$. It can be checked from the definition of $\simeq$ that this definition is independent of the pair (i, r) chosen.

It is routine to check that $DC_i = B_i$ $(i = 0, 1)$. Moreover, D is unique with this property since each C_i is epi. This completes the proof. ∎

The reader may like to check that the bound in Figure 4 is also a pushout.

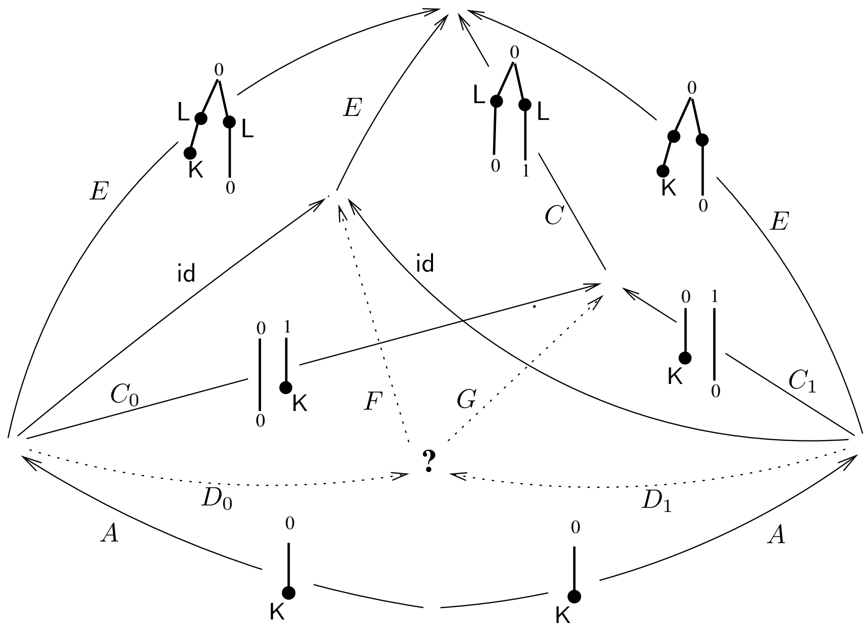


Fig. 5. RPOs do not always exist in abstract place graphs

Having developed the theory of place graphs far enough, we return to the question: how necessary is the notion of support? In particular, why should our s-category of place graphs pay respect to the identity of nodes? Let us see the effect on our theory if we neglect node-identity.

Following Definition 2.4, two place graphs are *support-equivalent* if they only differ by a bijection on their nodes. Taking the quotient of $\acute{\text{P}}\text{LG}_{\mathsf{h}}$ by support equivalence, we arrive at a category (not merely an s-category) of 'abstract' place graphs. In such place graphs, nodes with identical controls can no longer be distinguished. For example, the place graph C in Figure 5 has two nodes with control L which are not distinguishable. It follows that $CC_0 = CC_1 = E$; to see this, recall that the children of a root or node are unordered. This turns out to destroy the possibility of RPOs in general, as we now illustrate.

Example 6 (lack of RPOs) We claim that Figure 5 demonstrates the lack of RPOs (and hence the lack of pushouts) in abstract place graphs. In particular, it shows that no RPO exists for (A, A) to (E, E). For consider the two relative bounds $(\mathsf{id}, \mathsf{id}, E)$ and (C_0, C_1, C); from any RPO (D_0, D_1, D) there must exist mediating arrows F and G respectively to these two bounds. Now, from $FD_i = \mathsf{id}$ $(i = 0, 1)$ we deduce that D_0 and D_1 have no nodes; since they are hard place graphs we then find that $D_i = \mathsf{id}$ $(i = 0, 1)$. But then, since $GD_i = C_i$ $(i = 0, 1)$ we find that $C_0 = C_1$, a contradiction (recalling that a place graph is an *ordered* set of trees). ■

This example justifies the need for support, for RPOs to exist in place graphs. Similar counter-examples show the same need for link graphs, and hence for bigraphs.

7 Link graphs

Link graphs capture the connectivity of bigraphs, ignoring their nesting. There is a close formal analogy, but also differences, between the theories of place graphs and link graphs. Link graphs were presented fully in a previous paper [27]; the definitions and main results are repeated here to make the present paper self-contained.

As with place graphs, we assume a signature $\mathcal{K}$ assigning to each *control* K an arity $ar(K)$. We also assume an infinite set $\mathcal{X}$ of *names*.

Definition 7.1 (link graph) A *link graph* $A = (V, E, ctrl, link) : X \to Y$ has finite sets X of *inner names*, Y of *(outer) names*, V of *nodes* and E of *edges*. It also has a function $ctrl : V \to \mathcal{K}$ called the *control map*, and a function $link : X \uplus P \to E \uplus Y$ called the *link map*, where $P \stackrel{\text{def}}{=} \sum_{v \in V} ar(ctrl(v))$ is the set of *ports* of A.

We shall call the inner names X and ports P the *points* of A, and the edges E and outer names Y its *links*. The *support* of A is the set $V \uplus E$ of its nodes and edges.■

The outer and inner names are for interfacing, and will be important in defining composition. When we talk of a 'name' without adjective, we mean an outer name.

Definition 7.2 (idle, open, closed, peer, lean) A link is *idle* if it has no preimage under the $link$ map. Outer names are *open* links, edges are *closed* links. A point (i.e. an inner name or port) is *open* if its link is open, otherwise *closed*. Two distinct points are *peers* if they are in the same link. A link graph is *lean* if it has no idle edges. ■

Idle *names* play an important role; for example we may want to consider two bigraphs as members of the same homset, even if one of them uses a name x and the other does not. On the other hand idle *edges* serves no useful purpose, but may be created by composition. Sometimes we shall need to ensure that the property of leanness (no idle edges) is preserved by certain constructions.

Definition 7.3 (s-category of link graphs) The s-category $\acute{\text{L}}\text{IG}$ has name sets as objects and link graphs as arrows. The composition $A_1 A_0 : X_0 \to X_2$ of two link graphs

$$A_i = (V_i, E_i, ctrl_i, link_i) : X_i \to X_{i+1} (i = 0, 1)$$

is defined when their supports are disjoint; then $A_1 A_0 \stackrel{\text{def}}{=} (V, E, ctrl, link)$ where $V = V_0 \uplus V_1$, $ctrl = ctrl_0 \uplus ctrl_1$, $E = E_0 \uplus E_1$ and $link = (\mathsf{Id}_{E_0} \uplus link_1) \circ (link_0 \uplus \mathsf{Id}_{P_1})$. The identity link graph at X is $\mathsf{id}_X \stackrel{\text{def}}{=} (\emptyset, \emptyset, \emptyset_{\mathcal{K}}, \mathsf{Id}_X) : X \to X$.

The *tensor product* $\otimes$ in $\acute{\text{L}}\text{IG}$ is defined as follows: On objects, $X \otimes Y$ is simply the union of sets required to be disjoint. For two link graphs $A_i : X_i \to Y_i$ $(i = 0, 1)$ we take $A_0 \otimes A_1 : X_0 \otimes X_1 \to Y_0 \otimes Y_1$ to be defined when the interface products are defined and when A_0 and A_1 have disjoint node sets and edge sets; then to form their product we take the union of their link maps. ■

We can describe the composite link map $link$ of $A_1 A_0$ as follows, considering all possible arguments $p \in X_0 \uplus P_0 \uplus P_1$:

$$link(p) = \begin{cases} link_0(p) \text{ if } p \in X_0 \uplus P_0 \text{ and } link_0(p) \in E_0 \\ link_1(x) \text{ if } p \in X_0 \uplus P_0 \text{ and } link_0(p) = x \in X_1 \\ link_1(p) \text{ if } p \in P_1 \ . \end{cases}$$

By analogy with place graphs, we often denote the link map of A simply by A.

We have chosen to identify names in an interface alphabetically, not positionally. This difference is mathematically unimportant. Alphabetical names are convenient for link graphs just as they are convenient in the λ-calculus, and they also lead naturally to forms of parallel product that are familiar from process calculi.

Proposition 7.4 (isomorphisms in link graphs) *An arrow $\iota : X \to Y$ in ´LIG is an isomorphism iff it has no nodes or edges and its link map is a bijection from X to Y.*

There is an important variant of tensor product that merges outer names, i.e. does not require them to be disjoint. This has fewer algebraic properties than the tensor (categorically, it is not a bifunctor), but will be important in modelling process calculi:

Definition 7.5 (parallel product) The *parallel product* $\parallel$ in ´LIG is defined as follows: On objects, $X \parallel Y \stackrel{\text{def}}{=} X \cup Y$. On link graphs $A_i : X_i \to Y_i$ $(i = 0, 1)$ we define $A_0 \parallel A_1 : X_0 \otimes X_1 \to Y_0 \parallel Y_1$ whenever X_0 and X_1 are disjoint, by taking the union of link maps. ■

Now, analogous to place graphs, let us call a link graph *inner-injective* if no two inner names are peers. Then we can characterise epis and monos as follows:

Proposition 7.6 (epis and monos in link graphs) *A link graph is epi iff no name is idle; it is mono iff it is inner-injective.*

Notation When considering a pair $\vec{A} : W \to \vec{X}$ of link graphs with common domain W, we shall adopt a convention for naming their nodes, ports and edges. We denote the node set of A_i $(i = 0, 1)$ by V_i, and denote $V_0 \cap V_1$ by V_2. We shall use $v_i, v_i', \ldots$ to range over V_i $(i = 0, 1, 2)$. Similarly we use $p_i \in P_i$ and $e_i \in E_i$ for ports and edges $(i = 0, 1, 2)$. However, we shall sometimes use p_i also for points, i.e. $p_i \in W \uplus P_i$; the context will resolve any ambiguity. ■

As the reader will have noticed, there is a striking formal analogy between link graphs and place graphs. But the analogy is not complete. For a parent map is $prnt\colon h \uplus V \to V \uplus m$ where both the domain and codomain include the nodes V, while a link map is $link\colon W \uplus P \to E \uplus X$ where the sets P and E are disjoint; so unlike a parent map, a link map cannot be iterated, i.e. a link graph has no notion of *nesting*.

If we did not insist on working with *hard* place graphs, where there are no empty regions, then place graphs would possess RPOs but not, in general, pushouts; in that case the RPO theories for place graphs and link graphs are almost identical. The analogous 'hardening' of link graphs would be to require that no outer names are idle; in that case link graphs also have pushouts (when consistent). But here again the analogy fails; for in our intended applications it appears impossible to do without idle edges.

Thus we now embark upon an RPO theory for link graphs. Let us begin with some intuition. Suppose $\vec{D}$ is a bound for $\vec{A}$, and we wish to construct the RPO $(\vec{B}, B)$. To form $\vec{B}$, we shall first truncate $\vec{D}$ by removing outer names, and all points and edges not present in $\vec{A}$. Then for the outer face of $\vec{B}$, we create a new link (a name) for each point whose link was lost by the truncation, equating these new names only when required so that $B_0A_0 = B_1A_1$. Formally:

Construction 7.7 (RPOs in link graphs) An RPO $(\vec{B}\colon \vec{X} \to \hat{X}, B\colon \hat{X} \to Z)$, for a pair $\vec{A}\colon W \to \vec{X}$ of link graphs relative to a bound $\vec{D}\colon \vec{X} \to Z$, will be built in three stages. Since RPOs are preserved by isomorphism, we assume X_0, X_1 disjoint. We use the notational conventions introduced above.

nodes and edges: If V_i are the nodes of A_i ($i = 0, 1$) then the nodes of D_i are $(V_{\bar{\imath}} \setminus V_2) \uplus V_3$ for some V_3. Define the nodes of B_i and B to be $V_{\bar{\imath}} \setminus V_2$ ($i = 0, 1$) and V_3 respectively. Edges E_i are treated exactly analogously, and ports P_i inherit the analogous treatment from nodes.

interface: Construct the shared codomain $\hat{X}$ of $\vec{B}$ as follows. First, define the names in each X_i that must be mapped into $\hat{X}$:

$$X'_i \stackrel{\text{def}}{=} \{x \in X_i \mid D_i(x) \in E_3 \uplus Z\} .$$

Next, on the disjoint sum $X'_0 + X'_1$, define $\cong$ to be the smallest equivalence for which $(0, x_0) \cong (1, x_1)$ whenever $A_0(p_2) = x_0$ and $A_1(p_2) = x_1$ for some $p_2 \in W \uplus P_2$. Then define the codomain up to isomorphism:

$$\hat{X} \stackrel{\text{def}}{=} (X'_0 + X'_1)/\cong .$$

For each $x \in X'_i$ we denote the $\cong$-equivalence class of (i, x) by $\widehat{i, x}$.

links: Define B_0 to simulate D_0 as far as possible (B_1 is similar):

$$\text{For } x \in X_0 : \qquad B_0(x) \stackrel{\text{def}}{=} \begin{cases} \widehat{0,x} & \text{if } x \in X'_0 \\ D_0(x) & \text{if } x \notin X'_0 \end{cases}$$

$$\text{For } p \in P_1 \setminus P_2 : \ B_0(p) \stackrel{\text{def}}{=} \begin{cases} \widehat{1,x} & \text{if } A_1(p) = x \in X_1 \\ D_0(p) & \text{if } A_1(p) \notin X_1 \ . \end{cases}$$

Finally define B, to simulate both D_0 and D_1:

$$\text{For } \hat{x} \in \hat{X} : \ B(\hat{x}) \stackrel{\text{def}}{=} D_i(x) \ \text{ where } x \in X_i \text{ and } \widehat{i,x} = \hat{x}$$

$$\text{For } p \in P_3 : \ B(p) \stackrel{\text{def}}{=} D_i(p) \ .$$

■

This definition can be proved sound, i.e. the right-hand sides in the clauses defining the link maps B_i and B are well-defined links. Then the following is proved in [27]:

Theorem 7.8 (RPOs in link graphs) *In* $\grave{}\mathrm{LIG}$*, whenever a pair* $\vec{A}$ *of link graphs has a bound* $\vec{D}$*, Construction 7.7 yields an RPO* $(\vec{B}, B)$ *for* $\vec{B}$ *to* $\vec{D}$*.*

We now proceed to characterise all the IPOs for a given pair $\vec{A}: W \to \vec{X}$ of link graphs. We ask: how does our RPO $(\vec{B}, B)$ vary, when we keep $\vec{A}$ fixed but vary the given bound $\vec{D}$? As for place graphs, if $\vec{A}$ are both epi, then $\vec{B}$ remains fixed and only B varies, so that in this case there is a pushout. In $\acute{}\mathrm{PLG}_\mathsf{h}$ we confine ourselves to epis (since every hard place graph is epi), but for link graphs we need to treat the general case. The first step is to establish consistency conditions.

Definition 7.9 (consistency conditions for link graphs) We define three *consistency* conditions on a pair $\vec{A} : W \to \vec{X}$ of place graphs. We use p to range over arbitrary points, $p_i, p'_i, \ldots$ to range over P_i, and $p_2, p'_2, \ldots$ to range over $W \uplus P_2$, the shared points.

CL0 $ctrl_0(v_2) = ctrl_1(v_2)$

CL1 If $A_i(p) \in E_2$ then $p \in W \uplus P_2$ and $A_{\bar{\imath}}(p) = A_i(p)$.

CL2 If $A_i(p_2) \in E_i \setminus E_2$ then $A_{\bar{\imath}}(p_2) \in X_{\bar{\imath}}$, and if also $A_{\bar{\imath}}(p) = A_{\bar{\imath}}(p_2)$ then $p \in W \uplus P_2$ and $A_i(p) = A_i(p_2)$.

■

Again, let us express CL1 and CL2 in words. If $i = 0$, CL1 says that if the link of any point p in A_0 is closed and shared with A_1, then p is also shared and has the same link in A_1. CL2 says, on the other hand, that if the link of a shared point p_2 in A_0 is closed and *unshared*, then its link in A_1 must be open, and further that any peer of p_2 in A_1 must also be its peer in A_0.

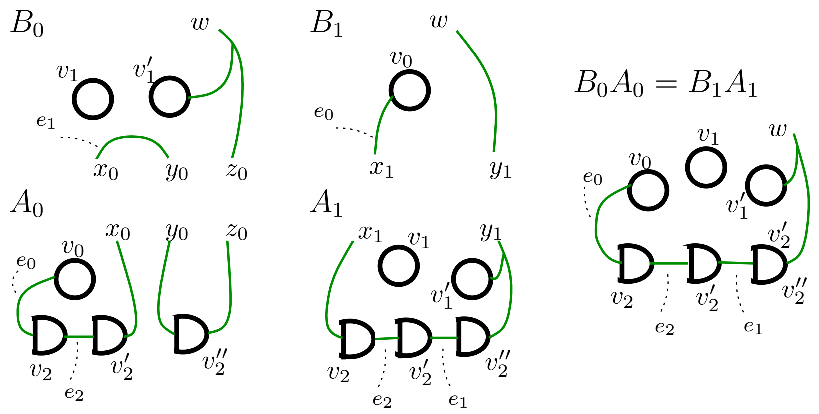


Fig. 6. A consistent pair $\vec{A}$ of link graphs, with bound $\vec{B}$

Proposition 7.10 (consistency in link graphs) *If the pair $\vec{A}$ has a bound, then the consistency conditions hold.*

Before going further, it may be helpful to see a simple example.

Example 7 (consistent link graphs) Consider the pair $\vec{A}\colon \emptyset \to \vec{X}$ of link graphs in Figure 6, where $X_0 = \{x_0, y_0, z_0\}$ and $X_1 = \{x_1, y_1\}$. Nodes with subscript 2 are shared. (Controls are not shown). The pair is consistent, with bound $\vec{B}$ as shown. It is worth checking the consistency conditions. ∎

Now, assuming the consistency conditions of Definition 7.9, for any given $\vec{A}$ we shall construct a non-empty family of IPOs. If $\vec{A}$ are both epi, then there is exactly one IPO up to isomorphism, and it is a pushout; the construction is close to that for place graphs. Otherwise the same construction yields an IPO, but further IPOs can be gained by *eliding* one or more of the idle names of A_i into C_i ($i = 0, 1$), i.e. the idle name can be incorporated into any of the edges of C_i. The choice of elisions —each yielding a different IPO— is represented below by the sets L_i and functions η_i ($i = 0, 1$).

Construction 7.11 (IPOs in link graphs) Assume the consistency conditions for the pair $\vec{A} : W \to \vec{X}$. We define a family of IPOs $\vec{C}\colon \vec{X} \to Y$ for $\vec{A}$ as follows.

nodes and edges: Take the nodes and edges of C_i to be $V_{\bar{i}} \setminus V_2$ and $E_{\bar{i}} \setminus E_2$.

interface: For $i = 0, 1$ choose any subset L_i of the names X_i such that all members of L_i are idle. Set $K_i = X_i \setminus L_i$. Define $K_i' \subseteq K_i$, the names to be mapped to the codomain Y, by

$$K_i' \stackrel{\text{def}}{=} \{x_i \in K_i \mid \forall p_2 \in W \uplus P_2.\ A_i(p_2) = x_i \Rightarrow A_{\bar{i}}(p_2) \in X_{\bar{i}}\} .$$

Next, on the disjoint sum $K_0' + K_1'$, define $\simeq$ to be the smallest equivalence such that $(0, x_0) \simeq (1, x_1)$ whenever $A_0(p_2) = x_0$ and $A_1(p_2) = x_1$ for some $p_2 \in W \uplus P_2$.

Then define the codomain up to isomorphism:

$$Y \stackrel{\text{def}}{=} (K'_0 + K'_1)/\simeq .$$

For each $x \in K'_i$ we denote the $\simeq$-equivalence class of (i, x) by $\widehat{i, x}$.

links: Choose two arbitrary functions $\eta_i : L_i \to E_{\bar{\imath}} \setminus E_2$ ($i = 0, 1$). Then define the link maps $C_i : X_i \to Y$ as follows (we give C_0; C_1 is similar):

For $x \in X_0$:

$$C_0(x) \stackrel{\text{def}}{=} \begin{cases} \widehat{0, x} & \text{if } x \in K'_0 \\ A_1(p_2) & \text{if } x \in K_0 \setminus K'_0, \text{ for some } p_2 \text{ with } A_0(p_2) = x \\ \eta_0(x) & \text{if } x \in L_0 \end{cases}$$

For $p \in P_1 \setminus P_2$:

$$C_0(p) \stackrel{\text{def}}{=} \begin{cases} \widehat{1, x} & \text{if } A_1(p) = x \in X_1 \\ A_1(p) & \text{if } A_1(p) \notin X_1 . \end{cases}$$ ■

The soundness of the above definition, and the fact that $\vec{C}$ is a bound, can both be routinely established.

Fortunately we shall not have to handle elisions in detail in this paper. It turns out that they are usually avoided in situations where we need to analyse an IPO. This can be either because the A_i in question has no idle names, or because the C_i in question has no edges (i.e. it is open).

The following characterisation theorem is proved in [27]:

Theorem 7.12 (characterising IPOs for link graphs) *A pair $\vec{C} : \vec{X} \to Y$ is an IPO for $\vec{A} : W \to \vec{X}$ iff it is generated (up to isomorphism) by Construction 7.11.*

8 Pure bigraphs: development

We now develop the theory of pure bigraphs. Proofs of propositions in this section can mostly be found in [23]. Several notions introduced here will be used in Part III for the dynamic theory.

First we combine the s-categories $\acute{\mathrm{P}}\mathrm{LG}_{\mathsf{h}}$ and $\acute{\mathrm{L}}\mathrm{IG}$:

Definition 8.1 (precategory of pure concrete bigraphs) The s-category $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\mathcal{K})$ of pure concrete bigraphs over a signature $\mathcal{K}$ has interfaces $I = \langle m, X \rangle$ as objects, with origin $\epsilon = \langle 0, \emptyset \rangle$, and bigraphs $G \colon I \to J$ as arrows. If $F : J \to K$ is another bigraph with $|F| \cap |G| = \emptyset$, then their composition is defined directly in terms of the compositions of the constituents as follows:

$$FG \stackrel{\text{def}}{=} \langle F^{\mathsf{P}} G^{\mathsf{P}}, F^{\mathsf{L}} G^{\mathsf{L}} \rangle \colon I \to K \ .$$

The identities are $\langle \mathsf{id}_m, \mathsf{id}_X \rangle : I \to I$, where $I = \langle m, X \rangle$. The tensor product of two interfaces is defined by $\langle m, X \rangle \otimes \langle n, Y \rangle \stackrel{\text{def}}{=} \langle m + n, X \uplus Y \rangle$ when X and Y are disjoint. The tensor product of two bigraphs $G_i : I_i \to J_i$ ($i = 0, 1$) with disjoint supports is defined as follows, when its interfaces are defined:

$$G_0 \otimes G_1 \stackrel{\text{def}}{=} \langle G_0^{\mathsf{P}} \otimes G_1^{\mathsf{P}}, G_0^{\mathsf{L}} \otimes G_1^{\mathsf{L}} \rangle \colon I_0 \otimes I_1 \to J_0 \otimes J_1 \ . \qquad \blacksquare$$

We shall omit the adjective ‘pure’ from now on. We shall also omit ‘concrete’ for the present; but in Definition 8.10 we shall introduce *abstract* bigraphs via a forgetful functor. We shall continue to omit the signature $\mathcal{K}$ except when it is important.

We now combine some familiar place graph and link graph structures:

Proposition 8.2 (isos, epis and monos in bigraphs) *A bigraph in* $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}$ *is iso (resp. epi, mono) iff its constituent place graph and link graph are both iso (resp. epi, mono).*

We shall call a bigraph *inner-injective* if both its place graph and its link graph are so. Thus a concrete bigraph is mono iff it is inner-injective. (The two properties differ for abstract bigraphs.)

We now observe that bigraphs are an instance of a structure from Section 3:

Proposition 8.3 (bigraphs are wide) $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\mathcal{K})$ *is a wide s-category. The interface* $I = \langle n, X \rangle$ *has* $\mathsf{width}(I) = n$*, and for* $G : \langle m, X \rangle \to \langle n, Y \rangle$ *the width map* $\mathsf{width}(G)$ *sends each site* $i \in m$ *to the unique root* $j \in n$ *such that* $i <_G j$.

It follows that when we later equip bigraphs with reaction rules we shall have a Wrs, and then we can apply the main congruence theorem, Theorem 4.6, provided that we have enough RPOs. So now we draw together our RPO results for place graphs and link graphs. We deduce from Theorem 6.9 and 7.8 the following:

Corollary 8.4 (RPOs for bigraphs) *In* $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}$ *an RPO for* $\vec{A}$ *to* $\vec{D}$ *is provided by*

$$(\, \langle B_0^{\mathsf{P}}, B_0^{\mathsf{L}} \rangle,\ \langle B_1^{\mathsf{P}}, B_1^{\mathsf{L}} \rangle,\ \langle B^{\mathsf{P}}, B^{\mathsf{L}} \rangle \,)$$

where $(\vec{B}^{\mathsf{P}}, B^{\mathsf{P}})$ *is an RPO for* $\vec{A}^{\mathsf{P}}$ *and* $(\vec{B}^{\mathsf{L}}, B^{\mathsf{L}})$ *is an RPO for* $\vec{A}^{\mathsf{L}}$ *to* $\vec{D}^{\mathsf{L}}$.

Similarly we deduce from Theorems 6.9 and 7.12 that:

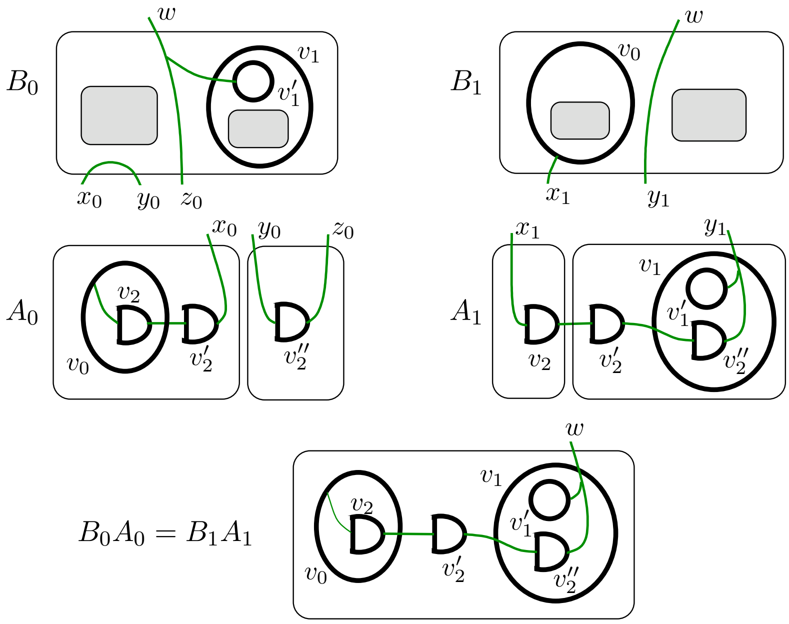


Fig. 7. A consistent pair $\vec{A}$ of bigraphs, with IPO $\vec{B}$

Corollary 8.5 (IPOs for bigraphs) *A pair $\vec{B}$ is an IPO for $\vec{A}$ in $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}$ iff $\vec{B}^{\mathsf{P}}$ is a place graph pushout for $\vec{A}^{\mathsf{P}}$ and $\vec{B}^{\mathsf{L}}$ is a link graph IPO for $\vec{A}^{\mathsf{L}}$.*

Example 8 (Bigraph IPOs) To illustrate IPOs in $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}$, we can combine Example 5 for place graphs and Example 7 for link graphs, since they have the same node sets. In both cases the bounds $\vec{B}$ are IPOs, and indeed pushouts because the graphs $\vec{A}$ are epi in this case. The combination is shown in Figure 7. ■

We now give a few special cases of IPOs. First, some pushouts (hence also IPOs):

Proposition 8.6 (containment pushout) *In any precategory, if A is epi then the pair (A, FA) has the pair (F, id) as a pushout. Hence, by taking $A = \mathsf{id}$ and $F = \mathsf{id}$ respectively:*

(1) Any pair (id, F) has (F, id) as a pushout.
(2) If A is epi then (A, A) has $(\mathsf{id}, \mathsf{id})$ as a pushout.

Next, tensor product preserves IPOs with disjoint support:

Proposition 8.7 (tensor IPO) *Let $\vec{C}$ be an IPO for $\vec{A}$ and $\vec{D}$ be an IPO for $\vec{B}$, where $|\vec{A}, \vec{C}| \cap |\vec{B}, \vec{D}| = \emptyset$. Then $(C_0 \otimes D_0, C_1 \otimes D_1)$ is an IPO for $(A_0 \otimes B_0, A_1 \otimes B_1)$, provided that all the interface products are defined.*

From this, with the help of Proposition 8.6, we deduce an important form of IPO. We shall need it in Appendix B, in the proof of Theorem 9.11, the adequacy theorem.

Corollary 8.8 (tensor IPOs with identities) *Let $A : I' \to I$ and $B : J' \to J$ have disjoint support, and let the names of I', I be disjoint from those of J', J. Then the pair $(A \otimes \mathsf{id}_{J'}, \mathsf{id}_{I'} \otimes B)$ has an IPO $(\mathsf{id}_I \otimes B, A \otimes \mathsf{id}_J)$. See diagram* (a).

In particular if $I' = J' = \epsilon$ then $A = a$ and $B = b$ are ground *bigraphs, and the IPO is as in diagram* (b).

(a)
$$\begin{tikzcd} I{\otimes}J' \arrow[r, "\mathsf{id}_I \otimes B"] & I{\otimes}J \\ {} \arrow[u, "A \otimes \mathsf{id}_{J'}"] \arrow[r, "\mathsf{id}_{I'} \otimes B"] & I'{\otimes}J \arrow[u, "A \otimes \mathsf{id}_J"'] \end{tikzcd}$$

(b)
$$\begin{tikzcd} I \arrow[r, "\mathsf{id}_I \otimes b"] & I{\otimes}J \\ {} \arrow[u, "a"] \arrow[r, "b"] & J \arrow[u, "a \otimes \mathsf{id}_J"'] \end{tikzcd}$$

We now prepare to define abstract bigraphs. In these, as promised, we forget the identity of nodes and edges, but we want to do a little more. Even without identity, idle edges may still lurk in a bigraph; we want to forget these too. Call a bigraph *lean* if its link graph is lean, i.e. has no idle edges. In Section 9 we shall need to transform IPOs by the addition or subtraction of idle edges. Let us write A^E for the result of adding a set E of fresh idle edges to A. The following is easy to prove from the IPO construction for link graphs:

Proposition 8.9 (IPOs, idle edges and leanness) *For any pairs $\vec{A}$ and $\vec{B}$ in $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}$:*

(1) If $\vec{B}$ is an IPO for $\vec{A}$, and A_1 is lean, then B_0 is lean.
(2) For any fresh set E of edges, $\vec{B}$ is an IPO for $\vec{A}$ iff (B_0, B_1^E) is an IPO for (A_0^E, A_1).

Definition 8.10 (abstract pure bigraphs and their category) Two concrete bigraphs A and B are *lean-support equivalent*, written $A \asymp B$, if after discarding any idle edges they are support equivalent. The category $\mathrm{BIG}_{\mathsf{h}}(\mathcal{K})$ of *abstract pure bigraphs* has the same objects as $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\mathcal{K})$, and its arrows are lean-support equivalence classes of concrete bigraphs. Lean-support equivalence is clearly a congruence (Definition 2.5). The associated quotient functor, assured by Definition 2.6, is

$$[\![\cdot]\!] : \acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\mathcal{K}) \to \mathrm{BIG}_{\mathsf{h}}(\mathcal{K}) \ . \qquad \blacksquare$$

Of course, there are also abstract versions of place graphs and link graphs. But we have little use for them, for we cannot combine an abstract place graph with an abstract link graph to form an abstract bigraph! (The combination only makes sense when nodes have identity.)

The reader might expect that we could henceforth develop our theory in abstract bigraphs, having constructed them. But this is impossible, since they lack RPOs — and even epis— in general. The lack of RPOs for abstract place graphs is shown by Example 6 in Section 6; other counterexamples can be found in [23]. However, the RPOs in *concrete* bigraphs will allow us in Section 9 to derive a behavioural congruence for ´$\text{B}\small{\text{IG}}_\mathsf{h}$; then we shall see how to transfer it, under certain assumptions, to $\text{B}\small{\text{IG}}_\mathsf{h}$.

We shall now introduce some notation and concepts used in following sections.

Notation We often abbreviate an interface $\langle 0, X\rangle$ to X, and $\{x\}$ to x; similarly we abbreviate $\langle m, \emptyset\rangle$ to m. Thus the interfaces $\emptyset$ and 0 are identical with the origin ϵ, and indeed the identity id_ϵ may be written variously as ϵ, $\emptyset$ or 0. ■

Although bigraphs are chiefly interesting for the nodes that generate their dynamic behaviour, node-free bigraphs are essential; they represent the ways in which bigraphs are wired together. A graph without nodes need have no sites or roots; this leads to the following definition:

Definition 8.11 (wiring, closure, substitution) A bigraph with interfaces of zero width, and hence having no nodes, is called a *wiring*; we let ω, ζ range over wirings. They are generated by composition and tensor product from two basic forms: $/x\colon x \to \epsilon$, called *closure*; and open wirings σ, τ which we call *substitutions*. We denote the empty substitution from ϵ to x by $x\colon \epsilon \to x$.

For $X = \{x_1, \ldots, x_n\}$ we write $/X$ for the multiple closure $/x_1 \otimes \cdots \otimes /x_n$, and X for the empty substitution $x_1 \otimes \cdots \otimes x_n$. For vectors $\vec{x}$ and $\vec{y}$ of equal length, with the x_i distinct, we write $\vec{y}/\vec{x}$ or $({}^{y_0}/{x_0}, {}^{y_1}/{x_1}, \ldots)$ for the surjective substitution $x_i \mapsto y_i$. Every substitution σ can be expressed uniquely as $\sigma = \tau \otimes X$, with τ surjective. We let α range over *renamings*, the bijective substitutions. ■

We now come to two kinds of bigraph *prime* and *discrete*, which are important both for the algebraic structure of bigraphs (Proposition 8.15) and for their dynamics (Definition 9.2). In both cases we are concerned with breaking down a bigraph into easy parts; for example, Proposition 8.15 will show that every bigraph is the composition of a wiring with a discrete bigraph.

Definition 8.12 (prime, discrete) An interface is *prime* if it has width 1. We shall often write a prime interface $I = \langle 1, X\rangle$ as $\langle X\rangle$; note in particular that $1 = \langle\emptyset\rangle$. A *prime* bigraph $P\colon m \to \langle X\rangle$ has no inner names and a prime outer face. An important prime is $\mathit{merge}\colon m \to 1$, where $m > 0$; it has no nodes, and simply maps m sites to a single root. A bigraph $G\colon m \to \langle n, X\rangle$ with no inner names is converted by merge into a prime $(\mathit{merge} \otimes \mathsf{id}_X)G$.

A bigraph is *discrete* if it has no edges, and its link map is bijective. Thus it is open, no two points are peers, and no name is idle. ■

Primes have no inner names; this ensures prime factorisation in Proposition 8.15.

Notation We often omit '$\ldots \otimes \mathsf{id}_I$' in compositions, when there is no ambiguity; for example we write $\mathit{merge}\, G$ for $(\mathit{merge} \otimes \mathsf{id}_X)G$.

Given a wiring $\omega\colon Y \to Z$, we may wish to apply it to a bigraph $G\colon I \to \langle m, X\rangle$ with fewer names, i.e. $Y = X \uplus X'$. Then we may write ωG for $(\mathsf{id}_m \otimes \omega)(G \otimes X')$ when m and X' can be understood from the context. Note that if ω closes a name in X' then ωG has a corresponding idle edge. ∎

We now look at variants of the tensor product, to reflect parallel composition in process calculi, for example $p \mid q$, which allow the processes p and q to share names. We first extend the parallel product of link graphs (Definition 7.5) as follows:

Definition 8.13 (parallel and prime product) The *parallel product* is defined on interfaces by $\langle m, X\rangle \parallel \langle n, Y\rangle \stackrel{\text{def}}{=} \langle m+n, X \cup Y\rangle$, and on bigraphs $G_i\colon I_i \to J_i$ $(i = 0, 1)$ with disjoint support by

$$G_0 \parallel G_1 \stackrel{\text{def}}{=} \langle G_0^{\mathsf{P}} \otimes G_1^{\mathsf{P}}, G_0^{\mathsf{L}} \parallel G_1^{\mathsf{L}}\rangle\colon I_0 \otimes I_1 \to J_0 \parallel J_1$$

when the interfaces exist. The *prime product* is defined on interfaces by $\langle m, X\rangle \mid \langle n, Y\rangle \stackrel{\text{def}}{=} \langle 1, X \cup Y\rangle$, and on bigraphs (under the same conditions) by

$$G_0 \mid G_1 \stackrel{\text{def}}{=} \mathit{merge}(G_0 \parallel G_1)\colon I_0 \otimes I_1 \to J_0 \mid J_1\ .$$ ∎

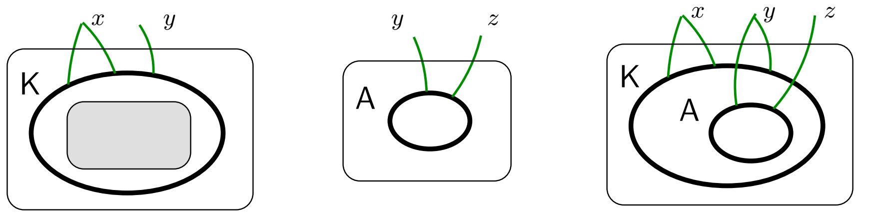
Both products are associative, and ϵ is the unit for $\parallel$. They are well-formed since the factors G_0 and G_1 are required to have disjoint inner names. The parallel product keeps their regions separate, while the prime product merges them. The notation $\mid$ comes from CCS and the π-calculus; the correspondence is accurate. Note that to join a wiring to a prime we may write either $\omega \mid P$ or $\omega \parallel P$; they coincide in this case.

We now introduce primitives involving nodes.

Definition 8.14 (ion, atom, molecule) Let K be a non-atomic control with arity k, and $\vec{x}$ a sequence of k names, not necessarily distinct. Let $X = \{\vec{x}\}$. We define the *ion* $K_{v,\vec{x}}\colon 1 \to \langle X\rangle$ to have a single K-node v, whose i^{th} port is linked to the name x_i for each $i \in k$. We omit the subscript v when it can be understood. For any prime P with names Y, the composite $(K_{\vec{x}} \parallel \mathsf{id}_Y)P\colon m \to \langle X \cup Y\rangle$ is a *molecule*; we shall abbreviate it to $K_{\vec{x}}.P$.

On the other hand if K is atomic it has no ion, but we define the *atom* $K_{\vec{x}}\colon \epsilon \to \langle X\rangle$; it resembles an ion but possesses no site.

Any closure of an ion is also an ion; similarly for molecules and atoms. ∎



the ion K_{xxy} the atom A_{yz} the molecule $K_{xxy}.A_{yz}$

The diagram shows a molecule built from an ion and an atom, in such a way that the ion and atom share a name. The possibility for nested nodes to share a name at different levels is important; our chosen notation $K_{\vec{x}}.P$ also agrees with the notation for prefixing in CCS and π-calculus. We shall see a close semantic correspondence in Section 11.

Let us now consider *discrete* bigraphs. In a precise sense they complement wiring:

Proposition 8.15 (discrete normal form) *Every bigraph $G\colon \langle m, X\rangle \to \langle n, Z\rangle$ can be expressed uniquely, up to a link iso on Y, as $G = (\mathsf{id}_n \otimes \omega)D$, where $\omega\colon Y \to Z$ is a wiring and $D\colon \langle m, X\rangle \to \langle n, Y\rangle$ is discrete. Furthermore every such discrete D may be factored uniquely, up to a place iso on the domain of each factor D_i, as*

$$D = \alpha \otimes ((D_0 \otimes \cdots \otimes D_{n-1})\iota)$$

with α a renaming, each D_i prime and discrete, and ι a permutation of sites.

Note that a renaming is discrete but not prime (since it has zero width); this explains α in the prime factorisation. Its uniqueness depends on the fact that primes have no inner names. In the special case that D is ground, the factorisation is just $D = d_0 \otimes \cdots \otimes d_{n-1}$, a product of prime discrete ground bigraphs.

The *discrete normal form (DNF)* applies equally to abstract bigraphs, and plays an important part in the complete axiomatisation of pure bigraphs [37]. Discreteness is well behaved in other ways. Clearly both composition and tensor product preserve it. IPOs also treat it well. In fact, we have:

Proposition 8.16 (properties of discreteness) *The discrete pure bigraphs form a sub-s-category of $\mathcal{B}\mathrm{IG}_{\mathsf{h}}$. Moreover*

(1) If D is discrete and (D', G') is an IPO for (G, D), then D' is discrete.
(2) If D, D' are discrete and $(D', \mathsf{id}_n \otimes \omega)$ bounds (G, D), then it is a pushout.

We have to make one more preparation for Section 9 on dynamics. When we define the notion of parametric reaction rule, we must allow a parametric redex to replicate some factors of its parameter and discard other factors. For example, the redex R for CCS shown in Figure 2 discards two of the four factors. We represent this by an operation $\overline{\eta}[\cdot]$ on parameters called *instantiation*. The following definition ensures that names are shared among all copies of a parameter factor.

Definition 8.17 (instantiation) Let $\eta\colon n \to m$ be a map of ordinals. For any $\acute{\mathbf{C}}$= $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\mathcal{K})$ and any X this defines an *instantiation* map

$$\overline{\eta}\colon \acute{\mathbf{C}}\langle m, X\rangle \to \acute{\mathbf{C}}\langle n, X\rangle$$

of ground arrows as follows. Decompose $g\colon \langle m, X\rangle$ into $g = \omega d$, where $d\colon \langle m, Y\rangle = d_0 \otimes \cdots \otimes d_{m-1}$, with each d_i prime and discrete. Then define

$$\overline{\eta}[g] \stackrel{\text{def}}{=} \omega(d'_0 \parallel \cdots \parallel d'_{n-1}) ,$$

where $d'_j \simeq d_{\eta(j)}$ for $j \in n$. This map is well-defined up to support translation, by Proposition 8.15. If η is injective, surjective or bijective then the instantiation $\overline{\eta}$ is said to be respectively *affine*, *total* or *linear*. ■

If $\overline{\eta}$ is not affine then it replicates at least one factor of the parameter d. Support translation is used to ensure that the several copies of a replicated factor have disjoint supports; also parallel product $\parallel$ is used because copies of a replicated factor will share names. If $\overline{\eta}$ is not total then the names of $d'_0 \parallel \cdots \parallel d'_{n-1}$ may be fewer than Y; this is how idle links may arise from reactions.

Proposition 8.18 (wiring an instance) *Wiring commutes with instantiation; that is,*

$$\zeta(\overline{\eta}[a]) \simeq \overline{\eta}[\zeta a] .$$

Proof Let $a\colon \langle m, X\rangle$, with $\eta : m' \to m$. Take the DNF $a = \omega d$, where $\omega\colon Y \to X$. Then $\overline{\eta}[a] = \omega d'$, where $d' = d'_0 \parallel \cdots \parallel d'_{m'-1}$ with each $d'_i \simeq d_{\eta(i)}$. So

$$\begin{aligned} \overline{\eta}[\zeta a] = \overline{\eta}[\zeta(\omega d)] \;&=\; \overline{\eta}[(\zeta\omega)d] \\ \simeq (\zeta\omega)d' \;&=\; \zeta(\omega d') \;\simeq\; \zeta(\overline{\eta}[a]) . \end{aligned}$$ ■

Part III : Dynamics for bigraphs

Section 9 introduces the notion of a *bigraphical reactive system* (Brs), which is an instance of the notion of Wrs from Part I. The dynamics of a Brs is provided by *parametric reaction rules*. Transition systems are set up, as defined in Part I; they are shown to yield congruential bisimilarity in both concrete and abstract Brss. A special class of *simple* Brss is defined; on the basis of work on Part I, it is shown that the standard transition system for a simple Brs can be significantly simplified. Section 10 introduces *sorted* Brss, in which (as in sorted algebras) the structure of bigraphs can be constrained in various ways to suit applications. It is shown that many *sortings* respect the dynamic theory. Finally, Section 11 illustrates every aspect of bigraphical theory in terms of a finite fragment of CCS, recovering exactly its original strong bisimilarity.

The concluding section discusses related and future work.

9 Reactions and transitions

We are now ready to apply our general notion of a wide reactive system (Wrs) to bigraphs. We begin this section by defining a *bigraphical reactive system* (Brs); we then discuss its standard transitions and show their induced bisimilarity is a congruence. Thereafter we specialise the results to the well-behaved subclass of *simple* Brss, where we can find a smaller transition system adequate for the standard one.

Bigraphical reactive systems

To define the notion of Brs, it remains to define reaction rules over bigraphs. We shall give a Brs a little more structure than a Wrs, since —as hinted in Section 3 and already illustrated for CCS in Figure 2 in the Introduction— we wish to identify the *parametric* reaction rules that will generate the ground rules of a Brs. First, let us define *activity* for bigraphs.

Definition 9.1 (active bigraph) A bigraph D is *active at* the site i if every node $>_D i$ has an active control. D is *active* if it is active at every site. ■

This defines the activity map for $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\mathcal{K})$ for any signature $\mathcal{K}$, and it is a routine matter to check that the conditions of Definition 3.3 hold.

For parametric reaction rules, we want a ground redex to have roughly the form $r = Rd$, where R is a parametric redex and d a parameter. But, since we are not dealing with name-binding, we wish the parameter's names to be also outer names of r; that is, R should not close them. We therefore choose parametric redexes to have the form $R\colon m \to J$, and for any parameter $d\colon \langle m, X\rangle$ we shall form a ground redex $r = (\mathsf{id}_X \otimes R)d$. Further, we shall use instantiations (Definition 8.17) to determine how a parameter should be instantiated. We arrive at the following:

Definition 9.2 (reaction rules for bigraphs) A *parametric reaction rule* has a *redex* R and *reactum* R', both lean. It takes the form

$$(\, R\colon m \to J,\; R'\colon m' \to J,\; \eta\,)$$

where $\eta\colon m' \to m$ is a map of ordinals. Then for every discrete $d\colon \langle m, X\rangle$ the parametric rule generates every ground reaction rule of the form (r, r'), where $r \doteq (\,(\mathsf{id}_X \otimes R)d$ and $r' \doteq (\mathsf{id}_X \otimes R')\,\overline{\eta}[d]$. We say that the rule is respectively *affine*, *total* or *linear* when the instantiation $\overline{\eta}$ is so. ■

Consider Example 2 in Section 1, displayed in Figure 2. In that case we have

$$R\colon 4 \to \langle x\rangle = \mathsf{alt}.(\mathsf{send}_x \mid \mathsf{id}) \mid \mathsf{alt}.(\mathsf{get}_x \mid \mathsf{id})\ , \qquad R'\colon 2 \to \langle x\rangle = x \mid \mathsf{id} \mid \mathsf{id}$$

and the instantiation, which is affine, is defined by $\eta\colon 0 \mapsto 0, 1 \mapsto 2$.

The reader may wonder why we choose parameters to be discrete. In fact the generated reaction relation would be unchanged if we allowed arbitrary ground bigraphs as parameters, since the instantiation of any ground bigraph is defined in terms of the factors of its underlying discrete bigraph. But discrete parameters simplify analysis considerably, especially for transitions and bisimilarity.

We are now ready to define our central concept:

Definition 9.3 (bigraphical reactive system) A *(concrete) bigraphical reactive system (Brs)* over $\mathcal{K}$ consists of $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\mathcal{K})$ equipped with a set $\acute{\mathcal{R}}$ of reaction rules closed under support equivalence ($\doteq$). We denote it by $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\mathcal{K}, \acute{\mathcal{R}})$. A Brs is respectively *affine*, *total* or *linear* when all its rules are so. ■

We have accented $\acute{\mathcal{R}}$, as well as $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}$, to indicate that our redexes and reacta are concrete. Now, since we have determined both the ground reaction rules and the activity of a Brs, we can assert that

Proposition 9.4 (a Brs is a Wrs) *Every bigraphical reactive system is a Wrs.*

We now turn to wide transition systems and bisimilarity. All of Section 4 on these topics can be applied to Brss, including the various transition systems such as FT and ST. Most importantly, from Theorem 4.6 we deduce a behavioural congruence:

Corollary 9.5 (congruence of wide bisimilarity) *In any concrete Brs with the standard transition system* ST*, wide bisimilarity* $\sim$ *is a congruence.*

Later we shall examine a particular class of Brss; it yields an adequacy theorem that significantly reduces the transition system ST. But first let us transfer our behavioural congruence to the abstract Brs $\mathrm{BIG}_{\mathsf{h}}(\mathcal{K},\mathcal{R})$, where $\mathrm{BIG}_{\mathsf{h}}(\mathcal{K})$ and $\mathcal{R}$ are obtained by the quotient functor $[\![\cdot]\!]$ of Definition 8.10.

This functor, the quotient by lean-support equivalence ($\Bumpeq$), is a little coarser than the quotient by support equivalence ($\bumpeq$). To transfer the congruence result we must first prove that $\Bumpeq$ respects ST:

Proposition 9.6 (transitions respect equivalence) *In a concrete Brs with* ST*:*

(1) Every transition label L is lean.
(2) Transitions respect lean-support equivalence ($\Bumpeq$) in the sense of Definition 4.2. That is, whenever $a \xrightarrow{L}_{\lambda} a'$, if $a \Bumpeq b$ and $L \Bumpeq M$ with Mb defined, then $b \xrightarrow{M}_{\lambda} b'$ for some b' such that $a' \Bumpeq b'$.
(3) Lean-support equivalence is a bisimulation.

Proof For part 1, use Proposition 8.9(1) and the fact that every discrete agent is lean. For part 2, use Proposition 8.9(2); the fact that each redex is lean ensures that it cannot share an idle edge with the agent a. Part 3 follows immediately from part 2. ∎

We are now ready to transfer the congruence results of Corollary 9.5 from concrete to abstract Brss. The following is immediate by invoking Theorem 4.8:

Corollary 9.7 (behavioural congruence in abstract Brss) *Let $\acute{\mathbf{C}}$ be a concrete Brs, and $\mathbf{C}$ its lean-support quotient. Let $\sim$ denote both the bisimilarity for* ST *in $\acute{\mathbf{C}}$ and the corresponding bisimilarity induced in $\mathbf{C}$. Then*

(1) $a \sim b$ iff $[\![a]\!] \sim [\![b]\!]$.
(2) Bisimilarity $\sim$ is a congruence in $\mathbf{C}$.

Simple Brss

The standard transition system ST is quite tractable, since each label is 'small' in the sense that it is the minimal required to allow a certain reaction. What chance do we have to reduce the transition set even further? One attempt would be to consider only those transitions of an agent a in which a makes a non-trivial contribution to the underlying reaction. To be precise, we would like to avoid considering any transition $a \xrightarrow{L}_{\lambda} a'$ arising from a redex r such that $|r| \subseteq |L|$.

We might expect such transitions to be redundant, because if a shares no node or edge with the redex then surely any other agent b should be able to make a transition with the same label; so to test whether $a \sim b$ we should not need to entertain such transitions.

It turn out that this intuition is correct, but only for some Brss. We now proceed to define the class of *simple prime* Brss, which (with one extra condition) will justify our intuition.

Recall from Section 7 that a link is *open* if it is a name, otherwise *closed*.

Definition 9.8 (simple, prime Brs) In $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}$ or $\mathrm{BIG}_{\mathsf{h}}$, call a bigraph *open* if every link is open. Call it *guarding* if it has no inner names, and no site has a root as parent. Call it *simple* [1] if it is inner-injective, open and guarding.

A Brs is *simple* (resp. *prime*) if all its redexes are simple (resp. prime). ∎

There are good arguments why we should expect redexes to satisfy the simpleness conditions; in fact, the only real constraint it imposes is in requiring a redex to be open. Even so, the reactive systems for CCS, π-calculus and mobile ambients all turn out to be simple.

We give without proof three easy properties of openness:

Proposition 9.9 (openness properties)

(1) A composition FG is open iff both F and G are open.
(2) Every open bigraph is lean (i.e. has no idle edges).
(3) If $\vec{B}$ is an IPO for $\vec{A}$ and A_1 is open, then B_0 is open.

We are now ready to define a sub-TS of the standard transition system, in which every transition of a is such that a makes a contribution to the underlying reaction:

Definition 9.10 (engaged transitions) In $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}$ a standard transition of a is said to be *engaged* if it can be based on a reaction with redex R such that $|a| \cap |R| \neq \emptyset$. Denote by $\acute{\mathrm{P}}\mathrm{E}$ the transition system of prime interfaces and engaged transitions. Denote by PE the sub-Lts in which the transitions are mono. ∎

We would like to prove PE is adequate for ST (Definition 4.9), i.e. that $\sim^{\mathrm{PE}}_{\mathrm{ST}} = \sim_{\mathrm{ST}}$ restricted to prime interfaces; then for prime a and b, to establish $a \sim_{\mathrm{ST}} b$ we need only prove $a \sim^{\mathrm{PE}}_{\mathrm{ST}} b$. For this we need only match each *engaged* transition of a (resp. b) by an arbitrary transition of b (resp. a). This is less work than matching *all* transitions. Note that the *relative* bisimilarity $\sim^{\mathrm{PE}}_{\mathrm{ST}}$ should not be confused with the *absolute* bisimilarity $\sim_{\mathrm{PE}}$. (They will be proved equal under certain conditions.)

For the engaged transitions to be adequate, it is not quite enough that the Brs should be simple and prime. [2] The following counter-example, due to Ole Jensen, shows

[1] This definition of 'simple' pertains only to pure bigraphs; a refined definition for binding bigraphs appears in [23]. Also, here we do not require a simple bigraph to be prime. We sometimes need primeness as well as simpleness, but it seems natural to separate the two notions.

[2] In the Technical Report [39] on which this paper is based it was wrongly claimed that these conditions were enough to ensure adequacy. I an grateful to Ole Jensen for pointing out the error.

what can go wrong.

Example 9 (engaged transitions not always adequate) Let $\mathsf{M}: 1$ and $\mathsf{N}: 0$ be atomic controls, and $\mathsf{L}: 0$ non-atomic. Consider the following two reaction rules:

$$\mathsf{L}.\, d \longrightarrow\!\!\triangleright\; d \mid d$$
$$\mathsf{M}_x \mid \mathsf{M}_x \longrightarrow\!\!\triangleright\; \mathsf{M}_x \;.$$

This defines a simple prime Brs, not affine. Consider two prime agents, $/x\, \mathsf{M}_x$ and N. Neither has an engaged transition, hence $/x\, \mathsf{M}_x \sim^{\text{PE}}_{\text{ST}} \mathsf{N}$. But each can do a unique L-transition, distinguishing them as follows (in which we omit the subscript $\lambda = 0$ from prime transitions):

$$/x\, \mathsf{M}_x \xrightarrow{\mathsf{L}} /x\, (\mathsf{M}_x \mid \mathsf{M}_x) \xrightarrow{\mathsf{id}} /x\, \mathsf{M}_x$$
$$\mathsf{N} \xrightarrow{\mathsf{L}} \mathsf{N} \mid \mathsf{N} \;\not\xrightarrow{\mathsf{id}}\; .$$

Thus engaged transitions are not adequate. ∎

There are two ways to ensure adequacy of engaged transitions. One way entails an alternative to name-closure, and involves *binding* bigraphs. It provides a satisfactory treatment of a wide range of Brss, including full CCS and the full π-calculus, and will be presented by Ole Jensen in his forthcoming PhD dissertation [21]. In this paper we limit our attention pure bigraphs; we shall therefore be content to ensure adequacy by imposing the affine condition. The resulting Lts for finite CCS will be given as illustration in Section 11.

The following theorem is proved in Appendix B.

Theorem 9.11 (adequacy of engaged transitions) *In a simple prime affine Brs equipped with* ST*, the prime engaged transitions are adequate; that is, engaged bisimilarity* $\sim^{\text{PE}}_{\text{ST}}$ *coincides with bisimilarity* $\sim_{\text{ST}}$ *on prime agents.*

In passing, we observe that simpleness and adequacy makes it easy to verify two desirable properties of idle names. Recall that a bigraph consisting of just an idle name x is itself denoted by x. Then adding extra idle names does not disturb bisimilarity; but also, it is possible that bisimilar agents may differ in their idle names. To be precise:

Proposition 9.12 (idle names and bisimilarity) *In a simple prime affine Brs equipped with* ST*,*

(1) $a \sim b$ *iff* $x \otimes a \sim x \otimes b$.
(2) $a \sim b$ *does not imply that* a *and* b *have the same idle names.*

Proof (1) For the forward implication, use congruence. For the converse we verify that $\mathcal{S} = \{(a, b) \mid x \otimes a \sim x \otimes b\}$ is a bisimulation. Let $a\mathcal{S}b$, and consider a transition $a \xrightarrow{L}_{\lambda} a'$. We easily deduce that $x \otimes a \xrightarrow{\mathsf{id}_x \otimes L}_{\lambda} x \otimes a'$, hence $x \otimes b \xrightarrow{\mathsf{id}_x \otimes L}_{\lambda} b''$ where $x \otimes a' \sim b''$. Assuming simpleness it can be shown that this transition of $x \otimes b$ cannot involve an elision of x. It is then easy to verify that b'' takes the form $x \otimes b'$ (up to isomorphism), where $b \xrightarrow{L}_{\lambda} b'$. But then $a'\mathcal{S}b'$ and we are done.

(2) Consider finite CCS with the rule of Example 2 in Section 1. Suppose it has at atomic control nil representing the null process. The agent $/x\, \mathsf{send}_x.\, \mathsf{send}_y.\, \mathsf{nil}$ attempts to send on the channel x, which is closed, and then to send on y. It has a single outer name y that is not idle. On the other hand the agent $y \otimes \mathsf{nil}$ has an idle name y. But neither agent has an engaged transition, so they are bisimilar. ■

We now wish to transfer PE to abstract Brss, via the quotient functor

$$[\![\cdot]\!] : \acute{}\mathrm{BIG}_{\mathsf{h}} \to \mathrm{BIG}_{\mathsf{h}}\ .$$

To do this, we would like to know that PE is *definite* for ST (see Definition 4.11), for then by Proposition 4.12 we can equate the relative bisimilarity $\sim^{\mathrm{PE}}_{\mathrm{ST}}$ with the absolute one $\sim_{\mathrm{PE}}$. For this, we need to know that, from the pair (L, λ) alone, we can determine whether or not a transition $a \xrightarrow{L}_{\lambda} a'$ is engaged.

It turns out that this holds in a wide range of Brss, including the natural encoding of π-calculus and ambient calculus. This is because they all satisfy a simple structural condition, namely that no rule subsumes another in the following sense:

Definition 9.13 (subsume) Define $\mathit{ctrl}(G)$, the *control* of a bigraph G, to be the multiset of controls of its nodes. Say that a rule with redex S *subsumes* another rule with redex R if $\mathit{ctrl}(R) \subsetneq \mathit{ctrl}(S)$. ■

Note that this property applies equally to concrete and abstract Brss; indeed a concrete Brs has a subsumption iff its image under the quotient functor $[\![\cdot]\!]$ has a subsumption. Now with the help of Corollary 4.13, we deduce

Corollary 9.14 (engaged congruence) *In a simple prime affine Brs with no subsumption:*

(1) The engaged transition system PE *is definite for* ST.
(2) Engaged bisimilarity $\sim_{\mathrm{PE}}$ *coincides with* $\sim_{\mathrm{ST}}$ *on prime agents.*
(3) For any context C *with prime interfaces,* $a \sim_{\mathrm{PE}} b$ *implies* $Ca \sim_{\mathrm{PE}} Cb$ *.*

We now proceed to transfer engaged transitions and bisimilarity from concrete to abstract bigraphs. Note that the term ‘engaged’ is defined only for concrete bigraphs; but for convenience we shall call an abstract transition *engaged* if it is the image under $[\![\cdot]\!]$ of an engaged transition; and we shall also call refer to the induced bisimilarity of abstract bigraphs as *engaged bisimilarity*.

Recall from Proposition 9.9 that every simple bigraph is lean. We derive the analogue of Corollary 9.7, with PE in place of ST, under extra assumptions:

Corollary 9.15 (engaged congruence in an abstract Brs) *Let* $\acute{\mathbf{C}}$ *be a simple prime affine Brs with no subsumption, and let* $\mathbf{C}$ *be its lean-support quotient. Let* $\sim_{\text{PE}}$ *denote bisimilarity both for* PE *in* $\acute{\mathbf{C}}$ *and for the induced transition system* $[\![\text{PE}]\!]$ *in* $\mathbf{C}$*. Then*

(1) $a \sim_{\text{PE}} b$ *iff* $[\![a]\!] \sim_{\text{PE}} [\![b]\!]$.
(2) In $\mathbf{C}$, $\sim_{\text{PE}}$ *is a congruence.*

Proof The quotient functor satisfies the conditions of Theorem 4.8. In particular, by Proposition 9.6 it respects PE, being a sub-Lts of ST. So the theorem yields part 1 immediately. It yields part 2 with the help of Corollary 9.14. ∎

Thus we have ensured congruence of engaged bisimilarity in an abstract Brs $\text{BIG}_{\mathsf{h}}(\mathcal{K},\mathcal{R})$ under reasonable assumptions.

10 Place sorting

In this short section we extend our results to certain *place-sorted* Brss, in which a sorting discipline constrains the parent map, thus limiting the admissible bigraphs. We begin with a brief motivation for sorting.

In significant applications we are likely to employ a rich signature, and to need some constraint on the way in which bigraphs may be built. For example, given a control K, we may want to constrain the children of a K-node to have only certain controls; or we may want to constrain the linkage allowed for some or all of the ports of a K-node. The latter kind of discipline we may call *link-sorting*; an instance of it was used in [27] for representing Petri nets. The former —the constraint on the parent map— we shall call *place-sorting*. Of course, we may combine link-sorting with place-sorting.

We cannot expect our bigraph theory to remain unaffected by sorting disciplines that place arbitrary constraints on the parent map and/or the link map. Such a constraint may prevent the existence of a tensor product, or of RPOs; or it may admit RPOs but affect their construction. We seek a broad definition of place-sorting disciplines, within which we can distinguish subclasses that respect our theoretical development, and which also reflect the demands of a wide range of applications.

We began this investigation in [35], and Jensen will continue it in his forthcoming

PhD Dissertation [21]. In order not to lengthen the present paper we shall not explore the full variety of sorting disciplines here. Instead we shall be content to give a very general definition of place-sorting, which can be done simply and briefly. Then, equally simply, we shall define one particular class of place-sorting disciplines — the *homomorphic* sortings — and assert that our main results concerning Ltss and congruential bisimilarity go through unchanged for Brss that obey these disciplines.

Thus we prepare the way for Section 11, encoding finite CCS in bigraphs. It provides a simple and effective example of homomorphic place-sorting, and the main result of this section will ensure that we shall derive a tractable Lts and a congruential bisimilarity for finite CCS.

In the following Θ will denote a non-empty set of *sorts*, and θ will range over Θ.

Definition 10.1 (place-sorted bigraphs) An interface $\langle m, X\rangle$ is Θ*-(place-)sorted* if it is enriched by ascribing a sort to each place $i \in m$. If I is place-sorted we denote its underlying unsorted interface by $\mathcal{U}(I)$.

We denote by ´BIG$_{\mathsf{h}}$($\mathcal{K}$,Θ) the s-category in which the objects are place-sorted interfaces, and each arrow $G\colon I \to J$ is a bigraph $G\colon \mathcal{U}(I) \to \mathcal{U}(J)$. The identities, composition and tensor product are as in ´BIG$_{\mathsf{h}}$($\mathcal{K}$), but with sorted interfaces. ■

Note that the width of an interface has been enriched from an ordinal m to a sequence in Θ^m. Thus it may be presented as $\langle\vec{\theta}, X\rangle$, where $\vec{\theta} = \theta_0, \ldots, \theta_{m-1}$; in particular a prime interface takes the form $\langle\theta, X\rangle$. Adding sorts to interfaces has, of course, done nothing to constrain the internal structure of bigraphs in ´BIG$_{\mathsf{h}}$($\mathcal{K}$,Θ), but has provided a means for adding such a constraint, as we now define:

Definition 10.2 (place-sorting) A *place-sorting* is a triple

$$\Sigma = (\mathcal{K}, \Theta, \Phi)$$

where Φ is a condition on the place graphs of Θ-sorted bigraphs over $\mathcal{K}$. The condition Φ must be satisfied by the identities and preserved by composition and tensor product.

A bigraph in ´BIG$_{\mathsf{h}}$($\mathcal{K}$, Θ) is Σ*-(place-)sorted* if it satisfies Φ. The Σ-sorted bigraphs form a sub-s-category of ´BIG$_{\mathsf{h}}$($\mathcal{K}$,Θ) denoted by ´BIG$_{\mathsf{h}}$(Σ). Further, if ´$\mathcal{R}$ is a set of Σ-sorted reaction rules then ´BIG$_{\mathsf{h}}$(Σ, ´$\mathcal{R}$) is a Σ*-sorted* Brs. ■

Even with only a single sort there are interesting examples, since Φ may impose constraints that have nothing to do with sorts. As a simple example, it may decree that each root or node has at most one child. (The reader may like to confirm that this sorting satisfies the required conditions.) As another example, it can represent atomicity of controls and nodes, by decreeing that nodes with certain controls have *no* children. Other examples, including the homomorphic sortings which we define shortly, are naturally expressed by first assigning a sort $\theta \in \Theta$ to every control, and then imposing constraints upon a bigraph in terms of the sorts thereby associated with nodes.

As already mentioned, arbitrary sortings may destroy our theory; for example, they may prevent the existence of RPOs, or prevent prime factorisation. What conditions must we place on a place-sorting $\Sigma = (\mathcal{K}, \Theta, \Phi)$ to ensure that our theory is preserved? This question can be addressed in terms of the forgetful functor which discards sorts:

$$\mathcal{U} : {}^{\acute{}}\mathrm{B{\small IG}}_{\mathsf{h}}(\Sigma) \to {}^{\acute{}}\mathrm{B{\small IG}}_{\mathsf{h}}(\mathcal{K}) \; .$$

We shall call $\mathcal{U}$ a *sorting* functor. Such functors have at least one property:

Proposition 10.3 (place-sorting is faithful) *On interfaces a sorting functor is surjective (but not in general injective). On each homset it is also* faithful*, i.e. injective (though not in general surjective) .*

However, we need more structure than this if we wish to apply our transition theory to a well-sorted Brs. An obvious example is that we may require the functor to preserve and/or reflect RPOs. We shall not continue this general investigation here. Instead, we shall focus on an attractive class of sortings. Such a sorting works by assigning a sort to each control, and hence to each node in a bigraph G (via its control map $ctrl_G$); the sorting also defining a parent function on sorts. The sorting condition requires, for each G, a homomorphism from the parent structure on its nodes to the parent structure on sorts. More precisely:

Definition 10.4 (homomorphic sorting) In a *homomorphic* sorting $\Sigma = (\mathcal{K}, \Theta, \Phi)$ the condition Φ assigns a sort $\theta \in \Theta$ to each control in $\mathcal{K}$. It also defines a parent map $prnt_\Sigma : \Theta \to \Theta$ over sorts.

In a bigraph G, via its control map, the sort assignment to $\mathcal{K}$ determines a sort for each node. Then Φ requires that, for each site or node w in G with sort θ:

– if $prnt_G(w)$ is a node then its sort is $prnt_\Sigma(\theta)$;

– if $prnt_G(w)$ is a root then its sort is θ. ∎

Homomorphic sorting is a tight discipline, in one important sense: by the second condition, the sort of every root (or outer region) in a well-sorted bigraph is uniquely determined by the sorts of its children, which must be identical. This, together with the first (homomorphic) condition, ensures that all the results in Sections 6, 8 and 9 hold for homomorphically sorted bigraphs. This refinement of the results involves, in all cases, no more than replacing all pure interfaces $\langle m, X\rangle$ by sorted interfaces $\langle \vec{\theta}, X\rangle$. We draw attention to some specific points.

The consistency conditions of Definition 6.6 are unchanged. As asserted in Proposition 6.7 they are necessary for a bound to exist, and they indeed ensure that Construction 6.8 constructs a (well-sorted) pushout for a pair of well-sorted place graphs. Thus, since place-sorting does not affect link graphs, the construction of RPOs for well-sorted bigraphs remains unchanged. The notions of primeness and discreteness remain unchanged, except that there is a prime interface $\langle\theta, X\rangle$ for each sort θ.

Parallel product remains unchanged, but prime product (Definition 8.13) is limited to bigraphs whose regions have identical sorts, since $merge\colon m \to 1$ is refined to $merge\colon (\theta, \ldots, \theta) \to \theta$. Discrete normal forms exist just as before. In an instantiation as defined by Definition 8.17, based on an ordinal map $\eta\colon n \to m$, there is an instantiation map $\overline{\eta}\colon \acute{\mathbf{C}}\langle\vec{\theta}, X\rangle \to \acute{\mathbf{C}}\langle\vec{\phi}, X\rangle$ whenever $\phi_j = \theta_{\eta(j)}$ $(j \in n)$.

All results on transitions, bisimilarity and congruence in the first part of Section 9 go through as before. In the second part, dealing with simple Brss, note that the properties *open*, *closed*, *guarding*, *prime*, *lean*, *simple*, *prime*, and *affine* mean for well-sorted bigraphs exactly what they mean for pure bigraphs, i.e. a well-sorted bigraph has the property iff its underlying pure bigraph has it. Finally, the adequacy theorem (Theorem 9.11) for engaged transitions holds; the proof, given in the Appendix, is virtually unchanged. We now restate it together with the important congruence result:

Theorem 10.5 (congruence and adequacy with sortings) *Let Σ be a homomorphic place-sorting. Then in $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\Sigma, \acute{\mathcal{R}})$:*

(1) $\sim_{\mathrm{ST}}$ is a congruence.
(2) If the reaction rules $\acute{\mathcal{R}}$ are simple prime affine, then PE *is adequate for* ST.

11 CCS revisited

We are now ready to see how our results apply to pure CCS [30], for which we gave a reaction rule in Example 2 of Section 1. This provides a nice application of the adequacy theorem and of place sorting, introduced in preceding sections.

We limit ourselves to finite pure CCS with the following syntax. We shall let P, Q range over *processes* and M, N over *sums* (or *alternations*); each summand of a sum is a process guarded by an action λ of the form x or $\overline{x}$.

$$\begin{aligned} P &::= \mathbf{0} \;\big|\; \nu x P \;\big|\; P \,|\, P \;\big|\; M \\ M &::= \lambda.P \;\big|\; M + M \\ \lambda &::= \overline{x} \;\big|\; x \quad . \end{aligned}$$

Restriction ν is the only scoping operator, and the free names of a process are just those not bound by ν. This is essentially the syntax of CCS as given in Definition 4.1 of [31], if $\mathbf{0}$ is taken to be the empty sum and process identifiers are omitted.

This syntax is two-sorted; we shall therefore translate it into a two-sorted s-category of abstract bigraphs $\mathrm{B{\small IG}}_{\mathsf{h}}(\Sigma_{\mathsf{ccs}})$, where $\Sigma_{\mathsf{ccs}} = (\mathcal{K}, \Theta, \Phi)$ is a homomorphic place sorting (Definition 10.4). The signature is

$$\mathcal{K} = \{\, \mathsf{nil}\colon 0\,,\ \mathsf{alt}\colon 0\,,\ \mathsf{send}\colon 1\,,\ \mathsf{get}\colon 1\,\}$$

and declares nil atomic, the other controls passive. We take $\Theta = \{\mathsf{p}, \mathsf{m}\}$, where p is for processes and m is for sums. The sorting condition Φ assigns the sort p to nil and alt, and the sort m to send and get; it also imposes the parent map $\{\mathsf{p} \mapsto \mathsf{m}, \mathsf{m} \mapsto \mathsf{p}\}$ on sorts. Recall also that each interface in $\mathrm{B{\small IG}}_{\mathsf{h}}(\Sigma_{\mathsf{ccs}})$ has sorted places, and that Φ imposes the homomorphic sorting conditions on the nodes, sites and roots of a bigraph.

We shall map CCS processes and sums into ground homsets with prime interfaces of the form $\langle \mathsf{p}, X\rangle$ and $\langle \mathsf{m}, X\rangle$. Thus we define two translation maps $\mathcal{P}_X[\cdot]$ and $\mathcal{M}_X[\cdot]$, each indexed by a finite name-set X, from finite pure CCS into $\mathrm{B{\small IG}}_{\mathsf{ccs}}$. These maps are defined whenever X includes all free names of the argument $(\cdot)$, so each process or sum has an image in many prime ground homsets.

Definition 11.1 (translation of finite CCS) The translations $\mathcal{P}_X[\cdot]$ for processes and $\mathcal{M}_X[\cdot]$ for sums are defined by mutual recursion:

$$\begin{array}{rl|rl}
\mathcal{P}_X[\mathbf{0}] = & X \mid \mathsf{nil} & \mathcal{M}_X[\overline{x}.P] = & \mathsf{send}_x.\mathcal{P}_X[P] \ \ (x \in X) \\
\mathcal{P}_X[\nu x P] = & /y\, \mathcal{P}_{y \uplus X}[\{y/x\}P] & \mathcal{M}_X[x.P] = & \mathsf{get}_x.\mathcal{P}_X[P] \quad (x \in X) \\
\mathcal{P}_X[P \mid Q] = & \mathcal{P}_X[P] \mid \mathcal{P}_X[Q] & \mathcal{M}_X[M{+}N] = & \mathcal{M}_X[M] \mid \mathcal{M}_X[N]\ . \\
\mathcal{P}_X[M] = & \mathsf{alt}.\,\mathcal{M}_X[M]\ . & & \blacksquare
\end{array}$$

In translating the prefix forms for input and output we have adopted the notation $\mathsf{K}_{\vec{x}}.P$ introduced in Definition 8.14, permitting names to be shared between an ion and its contents. In translating $\nu x P$ the name x is first alpha-converted to some $y \notin X$. We shall write $P \overset{\text{alpha}}{\equiv} Q$ to mean that P is alpha-convertible to Q. A substitution $\{y/x\}$ on CCS terms is metasyntactic, and not to be confused with the bigraph y/x.

Note that restriction and parallel composition are modelled directly by closure and prime product, and need no extra controls. It is perhaps surprising that summation '$+$' of CCS is also represented using prime product. But prime product in bigraphs is a purely structural or static operation, with no commitment to any dynamic interpretation. The distinction between parallel composition and summation in our bigraphical encoding of CCS is achieved by the form of its reaction rule, as we shall see.

Our translation maps are not injective on prime ground homsets. In fact they induce upon CCS an equivalence $\equiv$ that is close to the structural congruence defined in Definition 4.7 of [31]; the differences will be discussed shortly. But, due to sorting, they are *surjective*; this can be proved by induction on the number of nodes in a prime ground bigraph. We now make these points more precisely:

Definition 11.2 (structural congruence) Define *structural congruence* over CCS terms to be the smallest equivalence $\equiv$ that is preserved by all term constructions, and such that

(1) $P \overset{\text{alpha}}{\equiv} Q$ implies $P \equiv Q$, and $M \overset{\text{alpha}}{\equiv} N$ implies $M \equiv N$;

(2) '$\mid$' and '$+$' are associative and commutative under $\equiv$;

(3) $\nu x \nu y P \equiv \nu y \nu x P$;

(4) $\nu x P \equiv P$ and $\nu x\,(P \mid Q) \equiv P \mid \nu x Q$ for any x not free in P;

(5) $\nu x\, \lambda.P \equiv \lambda.\nu x P$ and $\nu x\,(M{+}\lambda.P) \equiv M + \lambda.\nu x P$

for any x not free in M or λ. ■

Note that clauses 4 and 5, taken in reverse, allow a restriction νx to be pulled outwards from any parallel component and any summand respectively. This gives rise to the following, analogous to the standard forms of Definition 4.8 in [31]:

Proposition 11.3 (CCS normal form) *Every CCS process is structurally congruent to a* normal form $\nu x_1 \cdots \nu x_\ell\, P$ *($\ell \geq 0$), where P is an* open process form *containing each name x_i free. Open process forms are defined recursively as follows:*

- *an* open process form *is a process term $P_1 \mid \cdots \mid P_m$ ($m > 0$), where each P_j is either* $\mathbf{0}$ *or an open sum form;*
- *an* open sum form *is a summation term $M_1{+}\cdots{+}M_n$ ($n > 0$), where each M_k takes the form $\lambda.P$ for some open process form P.*

These forms, with restrictions outermost, are important in proving the following theorem. It states essentially that each of our translation functions from CCS to bigraphs is a bijection from structural congruence classes to a prime ground homset:

Theorem 11.4 (bijective translation)

(1) The translations $\mathcal{P}_X[\cdot]$ and $\mathcal{M}_X[\cdot]$ are surjective on prime ground homsets.
(2) $P \equiv Q$ iff $\mathcal{P}_X[P] = \mathcal{P}_X[Q]$, and $M \equiv N$ iff $\mathcal{M}_X[M] = \mathcal{M}_X[N]$.

Proof (outline) For part 1 we prove, by induction on the number of nodes in each prime ground bigraph, that it has at least one preimage for the appropriate translation function.

For the forward implication of part 2 it is useful first to prove, by induction on the structure of process terms, that

$$P \overset{\text{alpha}}{\equiv} Q \text{ implies } \mathcal{P}_X[P] = \mathcal{P}_X[Q]$$
$$\text{and } M \overset{\text{alpha}}{\equiv} N \text{ implies } \mathcal{M}_X[M] = \mathcal{M}_X[N] \ ;$$

then the main property can be proved by a similar induction.

For the reverse implication of part 2 first observe that, by the forward implication, it will be enough to prove the result when P and Q are normal forms. For this, by considering the restrictions in P and Q, the task may be reduced to proving the property for open process forms. Finally, the property for open process forms and open sum forms can be proved by mutual induction on their structure. In this proof the crucial step is to show, in bigraphs, that if a_i $(i \in m)$ and b_j $(j \in n)$ are ground molecules such that

$$a_1 \mid \cdots \mid a_m = b_1 \mid \cdots \mid b_n \ ,$$

then $m = n$, and $a_i = b_{\pi(i)}$ for some permutation π on m. ∎

Having thus found an accurate representation for CCS terms up to structural congruence, we should point out two discrepancies between the latter and the standard version of structural congruence.

First, we do not have $P \mid \mathbf{0} \equiv P$; this is because we cannot encode $\mathbf{0}$ by the empty prime bigraph 1, since 1 is absent in hard bigraphs. This may seem to be a disadvantage of the latter. On the other hand hard bigraphs are easier to work with, as explained after Definition 6.1. Moreover, as we shall see, the *bisimilarity* $p \mid \mathsf{nil} \sim p$ holds.

The second discrepancy is clause 5 of Definition 11.2, which allows restriction to be pushed through an action prefix. In finite CCS this is a valid law. But in CCS with recursion (or replication), we cannot encode a restriction νx as name-closure in bigraphs, since this would not meet the requirement that every instance of a replicated process containing νx should have its own 'private copy' of x. Jensen [21] will present a proper encoding of restriction in such a case.

Now let us consider dynamics for $\text{B}\textsc{ig}_{\mathsf{ccs}}$. In our finite CCS we have the single reaction rule

$$(\overline{x}.P + M) \mid (x.Q + N) \longrightarrow P \mid Q \ ,$$

which may be applied anywhere not under an action prefix. On the other hand in $\text{B}\textsc{ig}_{\mathsf{ccs}}$ we have the reaction rule from Example 2, shown again here in Figure 8 with algebraic expressions for the redex R and reactum R'. It is easy to demonstrate that

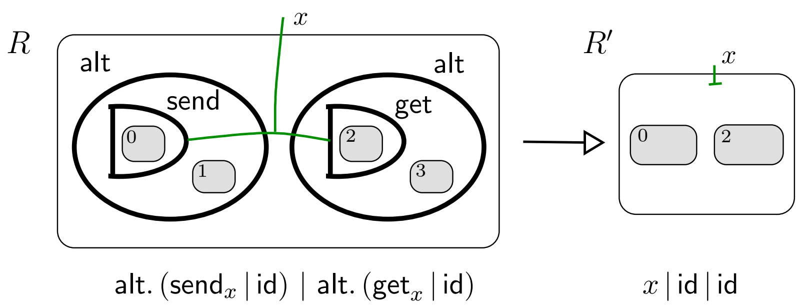


Fig. 8. The reaction rule (R, R', η) for BIG$_{\mathsf{ccs}}$

there is an exact match between the reaction relations generated in CCS and in BIG$_{\mathsf{ccs}}$, in the following sense:

Proposition 11.5 (matching reaction) $P \longrightarrow P'$ *iff* $\mathcal{P}_X[P] \longrightarrow\!\!\triangleright\, \mathcal{P}_X[P']$,

This exact match with CCS reaction has been achieved by working in abstract bigraphs. We now want to match with the original behavioural theory of CCS, which took the form of bisimilarity based upon a labelled transition system whose labels are not contexts. For the purpose of this comparison we have to digress into concrete bigraphs, since that is where we find the RPOs on which our contextual Ltss are based. So our starting point is the concrete sorted Brs

$$\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{ccs}} \stackrel{\text{def}}{=} \acute{\mathrm{B}}\mathrm{IG}_{\mathsf{h}}(\Sigma_{\mathsf{ccs}}, \acute{\mathcal{R}}_{\mathsf{ccs}}) \; ;$$

its reaction rules $(R, R', \eta) \in \acute{\mathcal{R}}_{\mathsf{ccs}}$ consist of all preimages (with R and R' lean) of the single abstract rule of BIG$_{\mathsf{ccs}}$ shown in Figure 8.

Our first step is to check that the prime engaged transitions in $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{ccs}}$ yield a congruential bisimilarity:

Corollary 11.6 (concrete bigraphical bisimilarity for CCS) *The bisimilarity* $\sim_{\text{PE}}$ *in* $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{ccs}}$ *is a congruence.*

Proof Since Σ_{ccs} is homomorphic we first deduce from the first part of Theorem 10.5 that $\sim_{\text{ST}}$ is a congruence; also from the second part, since $\acute{\mathcal{R}}_{\mathsf{ccs}}$ is simple prime affine, that PE is adequate for ST. Next note that $\acute{\mathrm{B}}\mathrm{IG}_{\mathsf{ccs}}$ has no subsumption, since it has only one rule. It follows from Corollary 9.14 that PE coincides with ST, and hence that $\sim_{\text{PE}}$ is a congruence. ∎

Now recall that we are using the term PE both for a concrete Lts and for its abstract image under the quotient by $[\![\cdot]\!]$. So, by analogy with Corollary 9.15, we are finally able to deduce congruential bisimilarity in our bigraphical representation of CCS:

Corollary 11.7 (abstract bigraphical congruences for CCS) *In* BIG$_{\mathsf{ccs}}$ *:*

(1) Two processes are bisimilar ($\sim_{\text{PE}}$) iff their concrete preimages are bisimilar.
(2) $\sim_{\text{PE}}$ is a congruence.

We devote the rest of this section to analysing this congruence. This will depend upon a structural analysis of the transitions in PE, and for this purpose we refer back to their preimages in ´BIG$_{\mathsf{ccs}}$, where we rely on the fact that they are engaged.

Every prime transition $p \xrightarrow{L} p'$ arises, then, from a ground rule (r, r') with redex

$$r = \mathsf{alt}.\,(\mathsf{send}_x.d\ \cdot\cdot) \mid \mathsf{alt}.\,(\mathsf{get}_x.e\ \cdot\cdot)$$

where '$\cdot\cdot$' stands for zero or more further factors in a discrete prime product, and (p, r) has (L, D) as an IPO with D active. Also p shares at least one of the nodes of the underlying parametric redex R: the two alt-nodes, the send-node and the get-node. What are the possibilities? Since p has sort p, if it shares the send-node then it must also share the parent alt-node; similarly if it shares the get-node. So there are two main sharing alternatives:

- p shares both nodes in one factor of R but none in the other;
- p shares all four nodes of R.

The former divides clearly into two symmetric cases. The latter also divides into two cases; either the send- and get-ports are joined by a closed link x, or they belong to possibly different open links.

	$p\colon I$	$L\colon I \to J$	$p'\colon J$	condition
1	$/Z\big(\mathsf{alt}.(\mathsf{send}_x.a\ \cdot\cdot) \mid b\big)$	$\mathsf{id}_I \mid \mathsf{alt}.(\mathsf{get}_x.c\ \cdot\cdot)$	$/Z(a \mid b) \mid c$	$x \notin Z$
2	$/Z\big(\mathsf{alt}.(\mathsf{get}_x.a\ \cdot\cdot) \mid b\big)$	$\mathsf{id}_I \mid \mathsf{alt}.(\mathsf{send}_x.c\ \cdot\cdot)$	$/Z(a \mid b) \mid c$	$x \notin Z$
3	$/Z\big(\mathsf{alt}.(\mathsf{send}_x.a_0\ \cdot\cdot)$ $\mid \mathsf{alt}.(\mathsf{get}_x.a_1\ \cdot\cdot) \mid b\big)$	id_I	$/Z(a_0 \mid a_1 \mid b)$	none
4	$/Z\big(\mathsf{alt}.(\mathsf{send}_x.a_0\ \cdot\cdot)$ $\mid \mathsf{alt}.(\mathsf{get}_y.a_1\ \cdot\cdot) \mid b\big)$	y/x	$/Z\, y/x$ $(a_0 \mid a_1 \mid b)$	$x \neq y$; $x, y \notin Z$

Fig. 9. The four forms for an engaged transition $p \xrightarrow{L} p'$

In Figure 9 we tabulate these four cases. It shows the structure of p, L and p' in each case, also taking account of the fact that —for a reaction to occur— any alt-node shared with R must occur actively in p. In the table, $a, b, c, \ldots$ stand for any processes, and '$\cdot\,\cdot$' for zero or more factors in a prime product; in the labels of cases 1 and 2 this product must be discrete. Note that, according to our convention, $^{y}/_{x}$ here denotes a substitution $\langle \mathsf{p}, X\rangle \to \langle \mathsf{p}, Y\rangle$, where $Y = (X \setminus x) \cup y$; its link map sends x to y and is otherwise the identity.

The reader will note that the expressions for labels in our table are *parametric*; for example in case 1, even for fixed p, there is a family of labels L, according as c and the unspecified factors '$\cdot\,\cdot$' vary. Moreover c reappears in the reactum p', whereas the factors '$\cdot\,\cdot$' are discarded. Such parametric labels were used by Hennessy, Merro and Zappa Nardelli [28,29] in their transition systems for mobile ambients. Jensen will show in his forthcoming PhD dissertation [21] that essentially the same labels are recovered in the bigraphical treatment of mobile ambients. So label-families, represented parametrically, seem to arise naturally when labels are taken to be contexts; they are not just an accidental consequence of our uniform approach to deriving Ltss. As we shall see shortly, they do not prevent analysis of the resulting behavioural relations.

Before discussing our derived Lts for CCS, let us establish a promised property:

Proposition 11.8 (unit for prime product) $p \sim p \,|\, \mathsf{nil}$.

Proof We shall prove the following relation to be a bisimulation:

$$\mathcal{S} \stackrel{\text{def}}{=} \{(p,\ p \,|\, \mathsf{nil}) \mid p \text{ an agent}\}\ .$$

First, suppose $p \xrightarrow{L} p'$ by the ground rule (r, r'); then the underlying IPO is as diagram (a) with $p' = Dr'$. Since none of the possible labels L (see Figure 9) guards its site, the IPO status is retained by adding a nil factor to both p and D, yielding an IPO as in (b), and thus $p \,|\, \mathsf{nil} \xrightarrow{L} (D \,|\, \mathsf{nil})r' = p' \,|\, \mathsf{nil}$, maintaining the relation $\mathcal{S}$.

(a)
$$\begin{tikzcd} {} \arrow[r, "L"] & {} \\ {} \arrow[u, "p"] \arrow[r, "r"] & {} \arrow[u, "D"'] \end{tikzcd}$$

(b)
$$\begin{tikzcd} {} \arrow[r, "L"] & {} \\ {} \arrow[u, "p \,|\, \mathsf{nil}"] \arrow[r, "r"] & {} \arrow[u, "D \,|\, \mathsf{nil}"'] \end{tikzcd}$$

In the other direction, suppose $p \,|\, \mathsf{nil} \xrightarrow{L} q'$ by the ground rule (r, r'); then in the underlying IPO, by commutation, the nil node cannot be shared by r, and indeed the IPO must be as in diagram (b), with $q' = (D \,|\, \mathsf{nil})r' = Dr' \,|\, \mathsf{nil}$. But the IPO status is retained by the omission of this shared nil-node, yielding an IPO as in diagram (a), so that we have $p \xrightarrow{L} Dr'$, again maintaining the relation $\mathcal{S}$.

This completes the proof. ∎

We are now ready to compare our derived transition system with the original CCS transitions, as presented in Part I of [31], which we shall call here the *raw* transitions; they use the non-contextual labels

$$\alpha \quad ::= \quad \overline{x} \;\Big|\; x \;\Big|\; \tau$$

where the first two represent sending and receiving a message, and τ represents a communication within the agent. Rather than reverting to CCS syntax, we set up the transitions $p \xrightarrow{\alpha} p'$ of this raw system directly in $\mathrm{B{\small IG}}_{\mathsf{ccs}}$; this will ease our comparison. The agents and label of each transition are characterised in Figure 10.

	p	α	p'	condition
1	$/Z\big(\mathsf{alt}.(\mathsf{send}_x.a\ \cdot\cdot) \mid b\big)$	$\overline{x}$	$/Z(a \mid b)$	$x \notin Z$
2	$/Z\big(\mathsf{alt}.(\mathsf{get}_x.a\ \cdot\cdot) \mid b\big)$	x	$/Z(a \mid b)$	$x \notin Z$
3	$/Z\big(\mathsf{alt}.(\mathsf{send}_x.a_0\ \cdot\cdot)$ $\mid \mathsf{alt}.(\mathsf{get}_x.a_1\ \cdot\cdot) \mid b\big)$	τ	$/Z(a_0 \mid a_1 \mid b)$	none

Fig. 10. The three forms for a raw transition $p \xrightarrow{\alpha} p'$

It can be seen that the raw transitions with these labels correspond closely to the first three forms shown in Figure 9; the notable difference is that, in the first two forms, the contextual label is composed with the agent, and the result of the transition is therefore larger than for the raw transitions.

However, there is no raw transition for the fourth (substitution) form of Figure 9; this is closely connected with the fact that the original CCS bisimilarity, which we shall denote here by $\sim_{\mathsf{ccs}}$, is not preserved by substitution. But the labels of the first three forms are mono, and the label ${}^{x}/{}_{y}$ of the fourth is not, since the names x, y are distinct and both occur in the agent's interface.

Now let us define *mono bisimilarity*, $\dot{\sim}$, to be the weakened version of standard bisimilarity ($\sim$) obtained by omitting the substitution labels from Figure 9. The above remarks suggest that $\dot{\sim}$ should coincide with the original CCS bisimilarity. We now verify this claim.

Theorem 11.9 (recovering CCS) *Mono bisimilarity recovers CCS, i.e.* $\dot{\sim} = \sim_{\mathsf{ccs}}$.

Proof ($\supseteq$) To show $\dot{\sim} \supseteq \sim_{\mathsf{ccs}}$ it will suffice to prove that

$$\mathcal{S} \stackrel{\text{def}}{=} \{(p_1|q, p_2|q) \mid p_1 \sim_{\mathsf{ccs}} p_2\}$$

is a bisimulation for $\dot{\sim}$; the result follows from Proposition 11.8 by taking $q = \mathsf{nil}$.

Assume $p_1 \sim_{\mathsf{ccs}} p_2$, and let $p_1 \mid q \xrightarrow{L} u_1$, where L is not a substitution label. We seek a transition $p_2 \mid q \xrightarrow{L} u_2$ such that $(u_1, u_2) \in \mathcal{S}$. We consider the cases for L; we need only consider cases 1 and 3 of Figure 9, since case 2 is like the first.

Case 1 $L = \mathsf{id} \mid \mathsf{alt}.(\mathsf{get}_x c \cdot\cdot)$. Then, from Figure 9, $p_1 \mid q$ contains an unguarded molecule $\mathsf{alt}.(\mathsf{send}_x a \cdot\cdot)$, in which x is free. There are two subcases:

If the molecule lies in q, then from Figure 9

$$\begin{aligned} q &= /Z(\mathsf{alt}.(\mathsf{send}_x a \cdot\cdot) \mid b) \\ u_1 &= p_1 \mid /Z(a \mid b) \mid c \end{aligned}$$

where $x \notin Z$ and we can assume no free name of p_2 lies in Z. Then, from Figure 9, $p_2 \mid q \xrightarrow{L} u_2 \stackrel{\text{def}}{=} p_2 \mid /Z(a \mid b) \mid c$. But $(u_1, u_2) \in \mathcal{S}$, so we are done.

On the other hand, if the molecule lies in p_1 then

$$\begin{aligned} p_1 &= /Z_1(\mathsf{alt}.(\mathsf{send}_x a_1 \cdot\cdot) \mid b_1) \\ u_1 &= /Z_1(a_1 \mid b_1) \mid q \mid c \end{aligned}$$

where $x \notin Z_1$ and we can assume no free name of q lies in Z_1. Then from Figure 10 there is a raw transition $p_1 \xrightarrow{\overline{x}} p_1' \stackrel{\text{def}}{=} /Z_1(a_1 \mid b_1)$, so $u_1 = p_1' \mid q \mid c$. But $p_1 \sim_{\mathsf{ccs}} p_2$, so for some p_2' we have $p_2 \xrightarrow{\overline{x}} p_2' \sim_{\mathsf{ccs}} p_1'$, and from Figure 10 we find

$$\begin{aligned} p_2 &= /Z_2(\mathsf{alt}.(\mathsf{send}_x a_2 \cdot\cdot) \mid b_2) \\ p_2' &= /Z_2(a_2 \mid b_2) \end{aligned}$$

where $x \notin Z_2$, and we can assume no free name of q or c lies in Z_2. Then from Figure 9 we find $p_2 \mid q \xrightarrow{L} u_2 \stackrel{\text{def}}{=} p_2' \mid q \mid c$. But $(u_1, u_2) \in \mathcal{S}$, so we are done.

Case 3 $L = \mathsf{id}$. Then $p_1 \mid q$ has an unguarded pair of molecules, together corresponding to a redex. There are four cases, depending on whether each molecule lies in p_1 or in q. If both lie in p_1 or both in q the argument is easy; we therefore consider just one of the remaining (symmetric) pair of cases.

Suppose then, consulting Figure 9, that

$$\begin{aligned} p_1 &= /Z_1(\mathsf{alt}.(\mathsf{send}_x a_1 \cdot\cdot) \mid b_1) \\ q &= /Z(\mathsf{alt}.(\mathsf{get}_x a \cdot\cdot) \mid b) \\ u_1 &= /Z_1(a_1 \mid b_1) \mid /Z(a \mid b) \end{aligned}$$

where we can assume that no free name of one is closed in the other, and $x \notin Z_1 \uplus Z$.

Then we have a raw transition $p_1 \xrightarrow{\overline{x}} p'_1 \stackrel{\text{def}}{=} /Z_1(a_1 \mid b_1)$. But $p_1 \sim_{\mathsf{ccs}} p_2$, so there exists p'_2 with $p_2 \xrightarrow{\overline{x}} p'_2 \sim_{\mathsf{ccs}} p'_1$, and by Figure 10 this takes the form

$$\begin{aligned} p_2 &= /Z_2(\mathsf{alt}.(\mathsf{send}_x a_2 \cdot \cdot) \mid b_2) \\ p'_2 &= /Z_2(a_2 \mid b_2)\ . \end{aligned}$$

Then from Figure 9 we deduce $p_2 \mid q \xrightarrow{\mathsf{id}} u_2 \stackrel{\text{def}}{=} p'_2 \mid /Z(a \mid b)$, and $(u_1, u_2) \in \mathcal{S}$, so we are done.

($\subseteq$) To show $\dot{\sim} \subseteq \sim_{\mathsf{ccs}}$ we shall prove that $\dot{\sim}$ is a bisimulation for $\sim_{\mathsf{ccs}}$. Assume $p \mathrel{\dot{\sim}} q$ and $p \xrightarrow{\alpha} p'$; we seek a matching transition $q \xrightarrow{\alpha} q'$ such that $p' \mathrel{\dot{\sim}} q'$.

If $\alpha = \overline{x}$ then the structure of p and p' is dictated by case 1 of Figure 10. Now, choosing $L = \mathsf{alt}.(\mathsf{get}_x.\mathsf{nil})$, we find from case 1 of Figure 9 that $p \xrightarrow{L} p' \mid \mathsf{nil}$. Since $p \mathrel{\dot{\sim}} q$ we have $q \xrightarrow{L} q''$ with $p' \mid \mathsf{nil} \mathrel{\dot{\sim}} q''$. By case 1 of both Figures 9 and 10 there exists q' such that $q'' = q' \mid \mathsf{nil}$ and $q \xrightarrow{\overline{x}} q'$. Appealing to Proposition 11.8, we then find $p' \mathrel{\dot{\sim}} q'$ as required.

The argument for $\alpha = x$ is exactly similar. The argument for $\alpha = \tau$ is even simpler, using case 3 of both Figures 9 and 10. This completes the proof of the theorem. ■

Having obtained a match in original CCS for mono bisimilarity $\dot{\sim}$, we also naturally try to match full derived bisimilarity $\sim$, which is stronger than mono bisimilarity. Since $\sim$ is preserved by all contexts, even substitutions, a natural candidate is *open bisimilarity*, as defined by Sangiorgi and Walker [47]. The latter is a good deal simpler for CCS than it is for the π-calculus [3]. Defined in BIG$_{\mathsf{ccs}}$ over the raw transition system, it consists of the smallest relation $\sim^{\circ}_{\mathsf{ccs}}$ such that, for all substitutions σ,

if $p \sim^{\circ}_{\mathsf{ccs}} q$ and $\sigma p \xrightarrow{\alpha} p'$, then $\sigma q \xrightarrow{\alpha} q'$ and $p' \sim^{\circ}_{\mathsf{ccs}} q'$ for some q'.

Since $\sim$ and $\sim^{\circ}_{\mathsf{ccs}}$ are both coinductively defined, it is relatively easy to compare them. In fact $\sim$ is strictly finer than $\sim^{\circ}_{\mathsf{ccs}}$. The proof of inclusion follows the lines of our proof that $\dot{\sim} \subseteq \sim_{\mathsf{ccs}}$. A counter-example to equality is provided by the pair

$$P = \nu z\big((\overline{x} + \overline{z}) \mid (y + z)\big) \qquad Q = \nu z\big((\overline{x}.y + y.\overline{x} + \overline{z}) \mid z\big)$$

where for convenience we use CCS notation, abbreviating $\lambda.\mathbf{0}$ to λ. This pair illustrates an interesting point. When translated into BIG$_{\mathsf{ccs}}$, P is has a transition labelled $^{x}/_{y}$; this can be seen as an 'observation' by P that its environment has connected the x-link with the y-link. On the other hand, Q has no such transition; so $P \not\sim Q$. But in the raw transition system such 'observations' are absent, and indeed $P \sim^{\circ}_{\mathsf{ccs}} Q$.

[3] This is because CCS agents never export restricted names ('scope extrusion').

This concludes our study of bigraphs applied to CCS, which has revealed considerable agreement with its original theory.

Discussion There are other equivalences for particular calculi that may or may not be matched by our approach based upon contextual labels. One example, examined in a previous paper [27], illustrates for Petri nets the phenomenon found here for CCS: that although derived contextual labels may differ from those in a raw Lts, the bisimilarity congruence may be matched exactly. Indeed, uniform methods may exist to simplify the derived contextual labels while retaining the induced bisimilarity, but this takes us beyond the scope of the present paper.

Moving aside from Ltss, can the bigraphical model contribute any uniform theory to the study of *barbed* bisimilarities [47]? Here the notion of labelled transition is replaced by unary predicates called barbs, written $\downarrow_{\mathsf{obs}}$, where obs indicates some observable property of states. The value of barbs (and their associated bisimilarity) is that they may be varied freely. For example, consider the asynchronous version of CCS, where the output action is just $\overline{x}$ rather than $\overline{x}.P$, i.e. the 'continuation' P is required to be empty. A useful barbed bisimilarity for this calculus is one in which the input action cannot be detected by a barb, i.e. the only barbs take the form $\downarrow_{\overline{x}}$ for detecting output actions. This illustrates that there are interesting behavioural relations that are not obviously expressible by means of Ltss.

Just as we can define a raw (non-contextual) Lts for any Brs (as we have done for one that represents CCS), so we can set up barbs for it, and hence study barbed bisimilarity within the uniform framework of bigraphs. Because barbed bisimilarities are useful and various, any uniform treatment of them will be valuable. However, they differ in one important way from those based upon Ltss. The latter are typically congruential (or nearly so); it has therefore been fruitful to explore a class of Ltss that *guarantee* congruence. On the other hand, barbed bisimilarities are not close to congruence in general; indeed, barbed congruence is typically defined *indirectly* as the largest congruence included in barbed bisimilarity. Thus the uniform study of barbed bisimilarity will require methods beyond those developed here; it is an open question whether such methods can be developed in bigraphs.

Finally, there are two directions in which the present methods may be extended. Leifer in his PhD dissertation [25] showed that, for transition systems based upon RPOs, certain more generous equivalences such as failures [5,20] and traces are uniformly congruential. Leifer also extended this treatment to *weak* bisimilarity, where 'silent' actions are ignored, and Jensen's forthcoming PhD dissertation will place this uniform treatment on a still broader footing.

12 Related and future work

We first turn to related work by other researchers, apart from those previously mentioned. The discussion then moves towards plans and ideas for future research.

Related work The longest tradition in graph reconfiguration —often called graph-rewriting— is based upon the *double pushout* (DPO) construction originated by Ehrig [13]. Our use of (relative) pushouts to derive transitions is quite distinct from the DPO construction, whose purpose is to explain the anatomy of graph-rewriting rules (not labelled transitions) working in a category of graph embeddings with graphs as objects and embeddings as arrows. This contrasts with our contextual s-categories, where objects are interfaces and arrows are bigraphs. But there are links between these formulations, both via cospans [16] and via a categorical isomorphism between graph embeddings and a coslice over s-categories [10]. Ehrig [14] has recently investigated these links further, after discussion with the author, and we believe that useful cross-fertilisation is possible. In the paper just cited, Gadducci, Heckel and Llabrés Segura [16] represent graph-rewriting by 2-categories, whose 2-cells correspond to our reactions. Several other formulations of graph-rewriting employ hypergraphs, for example Hirsch and Montanari [19]; their hypergraphs are not nested, but rewriting rules may replace a hyperedge by an arbitrary graph. Drewes *et al* [12] deal with hierarchical graphs, but their links do not join graphs at different levels.

Another use of 2-categories is by Sassone and Sobocinski [48]. They generalise RPOs to *groupoid* RPOs, in a 2-category whose 2-cells (i.e. arrows between arrows) are isomorphisms. They advocate treating dynamic entities (e.g. bigraphs) as arrows in such a 2-category. The 2-cells keep track of the identity of nodes, which is essential for RPOs to exist, and have the potential to serve as witnesses for rich structural congruences. An advantage in that approach over s-categories is that composition is total; a disadvantage is the more complicated notion of 2-RPO. Another advantage of 2-categories is that they lie closer to 'standard' category theory. However, the demands of our application are rather unlike those in other categorical applications; for example, it is essential —as our case studies have shown— to have a tractable analysis of the transitions based upon RPOs. Our s-categories lend themselves to this task; thus we shall retain them until some other approach eases analysis, such as the characterisation of IPOs and the adequacy theorem. In any case the 2-categorical approach is elegant and clearly deserves further development; the two approaches may then become complementary.

Concerning labelled transitions and bisimilarity, in recent work Merro and Hennessy [28] and Merro and Zappa Nardelli [29] have developed interesting labelled transition systems for the ambient calculus [8]; their labels are contextual. These appear to be the most detailed studies so far of behavioural equivalences for that calculus. As we now see, agreement with the bigraphical approach is becoming established.

Future work In his forthcoming PhD dissertation [21], Jensen develops bigraphical theory in a number of directions of intrinsic interest, which also support more refined case studies on behavioural analysis. First, he extends the work on weak bisimilarity begun by Leifer [25]. Second, he puts binding bigraphs (where names have scope) on a firmer footing than in [23], which gave their initial formulation. Third, he develops the theory of sorting, which was first used in [27] to encode Petri nets, and which was here illustrated in Sections 10 and 11. With these techniques, still deriving transition systems uniformly for Brss, he deals with the full π-calculus, and establishes a close match with the above-mentioned systems for ambients.

An important task for the immediate future is to stabilise and illustrate the treatment of *binding bigraphs*, where places are used to define the scope of links that represent bound names. Many applications require binding; the π-calculus is one, and Jensen [21] will treat this in his dissertation. We already know [23] that theory established for pure bigraphs can be simply transported to binding bigraphs, via a forgetful functor from the latter to the former. But there remain variants of binding, and the task is to establish the best choice and treat it thoroughly.

There is a large body of literature on rewriting systems and on the λ-calculus, comprehensively reported by Klop *et al* [51] and by Barendregt [1]. So far, the work on bigraphs has related chiefly to process calculi and their Ltss. It will be important to establish links with the tradition of rewriting systems. For example, the notion of confluence (of a rewriting system or some part of it) is likely to have very broad application in real-world pervasive and distributed systems, as discussed briefly below. To this end, scoped names in bigraphs are under further exploration [38]; it is found that multiple locality of names enables parametric reduction systems, such as the λ-calculus, to be represented succinctly. By this means, we hope that techniques for confluence —and other aspects of rewriting systems— can be lifted to bigraphs, where this broad range of applications can find expression.

One such application is to biological processes, already being explored by (for example) Cardelli [6], building on an original model by Shapiro *et al* [45,44] that used the π-calculus for this purpose. Cardelli has shown that more direct modelling is possible using the spatial quality of ambient-like reaction rules. But such experiments expose the need to adapt or extend spatially-aware models, like ambients and bigraphs, to accommodate real-world phenomena that lie beyond their present scope. One of these is a stochastic treatment of non-determinism, as developed in particular by Priami [43] for the π-calculus and used in the paper by Shapiro *et al*. Another important extension is to add the continuum and to allow continuous reactions. This is already done for the π-calculus by Rounds and Song [46] in the

Φ-calculus, which combines the mobility of the π-calculus with differential equations for the behaviour of real (i.e. continuous) variables. There is no barrier to this extension in bigraphs, since nothing in our formulation prevents a control signature from being denumerably infinite or even a continuum; for example, a family of controls indexed by the real numbers to represent distance. Of course there are technical hurdles to overcome —not least in the handling of infinitesimals.

Process theory also has strong tradition of non-standard logics such as temporal logic or the modal μ-calculus; these allow incremental analysis of processes, because simple properties (as opposed to full specifications) of a system can be expressed and verified one by one. For bigraphs, the obvious challenge is to find a logic that is *spatial* as well as temporal. Indeed, work by Caires and Cardelli on spatial logics for mobile ambients [7] has already been under way for a few years, and provides a promising starting point for a logic for bigraphs. A first step is taken in this direction by Conforti *et al* [11], where it can be seen that the independence of placing and linking leads to simplicity in the logical constructions.

Finally an initiative is being undertaken at the IT University in Copehagen, led by Lars Birkedal and Thomas Hildebrandt, in designing a bigraphical programming language (BPL) [4]. Two principal ideas are guiding this project: first, that programming and specification should arise out of sufficiently developed theory; second, that a practical language for experimental use in designing communicating systems is an essential vehicle for engineers to exert influence on further theoretical development.

Conclusion As we said at the outset, our model based on bigraphs is not definitive. Here and there we have made arbitrary choices, with the aim not only to generalise existing process theories but also to reach a level at which experiments can be mounted in describing and analysing real-world pervasive and distributed systems, both man-made and natural. Such experiments will certainly find shortcomings. By responding to these challenges, but retaining the continuity with existing calculi, we can aspire to a unity in process modelling that will truly justify preliminary efforts undertaken in computer science for more than three decades.

References

[1] Barendregt, H.P. (1981), *The Lambda Calculus.* Studies in Logic and the Foundations of Mathematics, Vol 103, North-Holland.

[2] Bergstra, J.A. and Klop, J.W. (1985), Algebra for communicating processes with abstraction. Theoretical Computer Science 37, pp77–121.

[3] Berry, G. and Boudol, G. (1992), The chemical abstract machine. Journal of Theoretical Computer Science, Vol 96, pp217–248.

[4] Birkedal, L. and Hildebrandt, T. (2004), Bigraphical programming languages. `LaCoMoCo.itu.dk`, Laboratory for Context-Dependent Mobile Communication, IT University, Denmark.

[5] Brookes, S.D., Hoare, C.A.R. and Roscoe, A.W. (1984), A theory of communicating sequential processes. J. ACM 31, pp560–599.

[6] Cardelli, L. (2003), Bioware languages. In: *Computer Systems: Theory, Technology and Applications*. Papers for Roger Needham, Springer Monographs in Computer Science, 6pp.

[7] Caires, L. and Cardelli, L. (2001), A spatial logic for concurrency (Part I). Proc. 4th International Symposium on Theoretical Aspects of Computer Software, LNCS 2215, Springer Verlag, pp1–37.

[8] Cardelli, L. and Gordon, A.D. (2000), Mobile ambients. Foundations of System Specification and Computational Structures, LNCS 1378, pp140–155.

[9] Castellani, I. (2001), Process algebras with localities. In: *Handbook of Process Algebra*, eds Bergstra, Ponse and Smolka, Elsevier, pp947–1045.

[10] Cattani, G.L., Leifer, J.J. and Milner, R. (2000), Contexts and Embeddings for closed shallow action graphs. University of Cambridge Computer Laboratory, Technical Report 496. [Submitted for publication.] Available at `http://pauillac.inria.fr/~leifer`.

[11] Conforti, G., Macedonio, D. and Sassone, V. (2005), Bilogics: spatial-nominal logics for bigraphs. Manuscript, available at `http://www.informatics.sussex.ac.uk/users/vs`.

[12] Drewes, F., Hoffmann, B. and Plump, D. (2000) Hierarchical graph transformation. In: *Foundations of Software Science and Computation Structures*, LNCS 1784, Springer Verlag.

[13] Ehrig, H. (1979), Introduction to the theory of graph grammars. In: *Graph Grammars and their Application to Computer Science and Biology*, LNCS 73, Springer Verlag, pp1–69.

[14] Ehrig, H. (2002), Bigraphs meet double pushouts. EATCS Bulletin 78, October 2002, pp72–85.

[15] Fournet, C. and Gonthier, G. (1996), The reflexive Cham and the join calculus. Proc. 23rd Annual ACM Symposium on Principles of Programming Languages, Florida, pp372–385.

[16] Gadducci, F., Heckel, R. and Llabrés Segura, M. (1999), A bi-categorical axiomatisation of concurrent graph rewriting. Proc. 8th Conference on Category Theory in Computer Science (CTCS'99), Vol 29 of Electronic Notes in TCS, Elsevier Science.

[17] Gardner, P.A. (2000), From process calculi to process frameworks. Proc. CONCUR 2000, 11th International Conference on Concurrency Theory, pp69–88.

[18] Gardner, P.A. and Wischik, L. (2000), Explicit fusions. Proc. MFCS 2000. LNCS 1893, pp373–383.

[19] Hirsch, D. and Montanari, U. (2001), Synchronised hyperedge replacement with name mobility. Proc. 12th International Conference on Concurrency Theory (CONCUR 2001), LNCS 2154, pp121–136.

[20] Hoare. C.A.R. (1985), *Communicating Sequential Processes.* Prentice Hall.

[21] Jensen, O.H. (2005), Forthcoming PhD Dissertation.

[22] Jensen, O.H. and Milner, R. (2003), Bigraphs and transitions. In 30th SIGPLAN-SIGACT Symposium on Principles of Programming Languages.

[23] Jensen, O.H. and Milner, R. (2004), Bigraphs and mobile processes (revised). Technical Report UCAM-CL-TR-580, University of Cambridge Computer Laboratory.

[24] Lafont, Y. (1990), Interaction nets. Proc. 17th ACM Symposium on Principles of Programming Languages (POPL'90), pp95–108.

[25] Leifer, J.J. (2001), Operational congruences for reactive systems. PhD Dissertation, University of Cambridge Computer Laboratory. Distributed in revised form as Technical Report 521. Available from `http://pauillac.inria.fr/~leifer`. (to appear). in Submitted for publication.

[26] Leifer, J.J. and Milner, R. (2000), Deriving bisimulation congruences for reactive systems. Proc. CONCUR 2000, 11th International Conference on Concurrency theory, pp243–258. Available at `http://pauillac.inria.fr/~leifer`.

[27] Leifer, J.J. and Milner, R. (2004), Transition systems, link graphs and Petri nets. Technical Report 598, University of Cambridge Computer Laboratory. Revised version to appear Mathematical Structures in Computer Science. Available from `http://www.cl.cam.ac.uk/users/rm135`.

[28] Merro, M. and Hennessy, M. (2002), Bisimulation Congruences in Safe Ambients. Proc. 29th International Symposium on Principles of Programming Languages, Oregon, pp71–80.

[29] Merro, M. and Zappa Nardelli, F. (2003), Bisimulation proof methods for mobile ambients. In Proc. ICALP 2003. Springer Verlag.

[30] Milner, R. (1980), *A Calculus of Communicating Systems.* LNCS 92, Springer Verlag.

[31] Milner, R. (1999), *Communicating and Mobile Systems: the π-calculus.* Cambridge University Press.

[32] Milner, R. (1996), Calculi for interaction. Acta Informatica 33, pp707–737.

[33] Milner, R. (2001), Computational flux. Proc 28th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages, pp220-221.

[34] Milner, R. (2001), Bigraphical reactive systems. Proc. 12th International Conference on Concurrency Theory, LNCS 2154, pp16–35.

[35] Milner, R. (2001), Bigraphical reactive systems: basic theory. Technical Report 503, University of Cambridge Computer Laboratory. Available from `http://www.cl.cam.ac.uk/users/rm135`.

[36] Milner, R. (2004), Bigraphs for Petri nets. Proc. Advanced course in Petri nets (Eichstaett '03), LNCS 3098.

[37] Milner, R. (2004), Axioms for bigraphical structure. Mathematical Structures in Computer Science (to appear). Available from `http://www.cl.cam.ac.uk/users/rm135`.

[38] Milner, R. (2005), Bigraphs whose names have multiple locality. Technical Report TR603, University of Cambridge Computer Laboratory. Available from `http://www.cl.cam.ac.uk/users/rm135`.

[39] Milner, R. (2005), Pure bigraphs. Technical Report 614, University of Cambridge Computer Laboratory.

[40] Milner, R., Parrow, J. and Walker D. (1992), A calculus of mobile processes, Parts I and II. Journal of Information and Computation, Vol 100, pp1–40 and pp41–77.

[41] Park, D.M.R. (1980), Concurrency and automata on infinite sequences. LNCS 104, Springer Verlag.

[42] Parrow, J. and Victor, B. (1998), The fusion calculus: expressiveness and symmetry in mobile processes. Proc. LICS'98, IEEE Computer Society Press.

[43] Priami, C. (1995), Stochastic π-calculus. Computer Journal, Vol 38 (7), pp578–589.

[44] Priami, C., Regev, A., Silverman, W. and Shapiro, E. (2001), Application of stochastic process algebras to bioinformatics of molecular processes. Information Processing Letters, Vol 80, pp 25–31.

[45] Regev, A., Silverman, W. and Shapiro, E. (2001), Representation and simulation of biochemical processes using the π-calculus process algebra. Proc. Pacific Symposium of Biocomputing 2001 (PSB2001), Vol 6, pp459–470.

[46] Rounds, W.H. and Song, H. (2003), The Φ-calculus - a language for distributed control of reconfigurable embedded systems. In: *Hybrid Systems: Computation and control*, LNCS 2263, Springer-Verlag, pp435–449.

[47] Sangiorgi, D. and Walker, D. (2001), *The π-calculus*. Cambridge University Press.

[48] Sassone, V. and Sobocinski, P. (2002), Deriving bisimulation congruences: a 2-categorical approach. Electronic Notes in Theoretical Computer Science, Vol 68 (2), 19pp.

[49] Sewell, P. (1998), From rewrite rules to bisimulation congruences. Proc CONCUR'98, LNCS 1466, pp269–284. Full version to appear in Theoretical Computer Science, Vol 272/1–2.

[50] Smith, H. and Fingar, P. (2002), *Business Process Management: the third wave*. Amazon.com.

[51] Terese, (2003), *Term Rewriting Systems.* Cambridge University Press.

[52] Wojciechowski, P.T. and Sewell, P. (1999), Nomadic Pict: Language and infrastructure design for mobile agents. Proc. ASA/MA '99, Palm Springs, California.

APPENDIX

A Induced congruences

In this appendix we prove Theorem 4.8, which ensures that certain quotient functors preserve and reflect bisimilarity and preserve its congruence.

Theorem 4.8 (transitions induced by functors) *Let $\acute{\mathbf{C}}$ be equipped with an Lts $\mathcal{L}$. Let $\mathcal{F}\colon \acute{\mathbf{C}} \to \acute{\mathbf{D}}$ be a full Wrs functor that is the identity on objects and respects $\mathcal{L}$, and such that $\mathcal{F}(a) = \mathcal{F}(b)$ whenever $a \bumpeq b$. Then the following hold for $\mathcal{F}(\mathcal{L})$:*

(1) $a \sim b$ in $\acute{\mathbf{C}}$ iff $\mathcal{F}(a) \sim \mathcal{F}(b)$ in $\acute{\mathbf{D}}$.
(2) If bisimilarity is a congruence in $\acute{\mathbf{C}}$ then it is a congruence in $\acute{\mathbf{D}}$.

Proof 1. ($\Rightarrow$) We establish in $\acute{\mathbf{D}}$ the bisimulation

$$\mathcal{R} = \{(\mathcal{F}(a), \mathcal{F}(b)) \mid a \sim b\} .$$

Let $a \sim b$ in $\acute{\mathbf{C}}$, and let $p = \mathcal{F}(a)$, $q = \mathcal{F}(b)$ and $p \xrightarrow{M}\!\triangleright_{\lambda} p'$ in $\acute{\mathbf{D}}$. Then by definition of the induced Lts the triple (p, M, p') has an $\mathcal{F}$-preimage (a_1, L, a_1') such that $a_1 \xrightarrow{L}\!\triangleright_{\lambda} a_1'$ in $\acute{\mathbf{C}}$; moreover, since $L \bumpeq L' \Rightarrow \mathcal{F}(L) = \mathcal{F}(L')$, L can be chosen so that La and Lb are defined. Since $\mathcal{F}$ respects $\mathcal{L}$ there exists a' with $p' = \mathcal{F}(a')$ such that $a \xrightarrow{L}\!\triangleright_{\lambda} a'$.

Since $a \sim b$ and Lb is defined, there exists b' such that $b \xrightarrow{L}\!\triangleright_{\lambda} b'$ and $a' \sim b'$. It follows that $q \xrightarrow{M}\!\triangleright_{\lambda} q'$ in $\acute{\mathbf{D}}$, where $q' = \mathcal{F}(b')$ and $(p', q') \in \mathcal{R}$, so we are done.

1. ($\Leftarrow$) We establish in $\acute{\mathbf{C}}$ the bisimulation

$$\mathcal{S} = \{(a, b) \mid \mathcal{F}(a) \sim \mathcal{F}(b)\} .$$

Let $\mathcal{F}(a) \sim \mathcal{F}(b)$ in $\acute{\mathbf{D}}$, and let $p = \mathcal{F}(a)$, $q = \mathcal{F}(b)$ where $a \xrightarrow{L}\!\triangleright_{\lambda} a'$ in $\acute{\mathbf{C}}$ with Lb defined. Then $p \xrightarrow{M}\!\triangleright_{\lambda} p'$ in $\acute{\mathbf{D}}$, where $M = \mathcal{F}(L)$ and $p' = \mathcal{F}(a')$. So for some q' we have $q \xrightarrow{M}\!\triangleright_{\lambda} q'$ with $p' \sim q'$.

This transition must arise from a transition $b_1 \xrightarrow{L_1}\!\triangleright_{\lambda} b_1'$ in $\acute{\mathbf{C}}$, where $q = \mathcal{F}(b_1)$, $M = \mathcal{F}(L_1)$ and $q' = \mathcal{F}(b_1')$. But then $b_1 \equiv b$ and $L_1 \equiv L$, where $\equiv$ is the equivalence induced by $\mathcal{F}$; we also have Lb defined, and $\mathcal{L}$ respects $\equiv$, so we can find b' for which $b \xrightarrow{L}\!\triangleright_{\lambda} b'$ and $b_1' \equiv b'$. But $(a', b') \in \mathcal{S}$, so we are done.

2. Assume that bisimilarity in $\acute{\mathbf{C}}$ is a congruence. In $\acute{\mathbf{D}}$, let $p \sim q$ with $p, q\colon I$, and let $G\colon I \to J$ be a context with Gp and Gq defined. Then since $\mathcal{F}$ is full there exist

$a, b \colon I$ and $C \colon I \to J$ in $\acute{\mathbf{C}}$ with $p = \mathcal{F}(a)$, $q = \mathcal{F}(b)$ and $G = \mathcal{F}(C)$; moreover, since $C \bumpeq C' \Rightarrow \mathcal{F}(C) = \mathcal{F}(C')$, C can be chosen so that Ca and Cb are defined.

From 1.($\Leftarrow$) we have $a \sim b$, hence by assumption $Ca \sim Cb$. Applying the functor $\mathcal{F}$ we have from 1.($\Rightarrow$) that $Gp \sim Gq$ in $\acute{\mathbf{D}}$, as required. ■

B The adequacy theorem

This appendix proves Theorem 9.11, asserting the adequacy of engaged transitions for prime agents in a simple prime affine Brs.

We begin with a proposition asserting that if we apply an affine instantiation $\overline{\eta}$ to a composition Ga, where a is prime, then the form of the result is independent of a.

Proposition B.1 (affine instantiation) *Let $G \colon \langle X \rangle \to \langle m, Z \rangle$ be a context, and $\eta :: n \to m$ an injective map. Then one of the following holds:*

$$\textit{either there exists } C \colon \langle X \rangle \to \langle n, Z \rangle \textit{ such that } \overline{\eta}[Ga] \bumpeq Ca \textit{ for all } a\ ;$$

$$\textit{or there exists a ground } c \colon \langle n, Z \rangle \textit{ such that } \overline{\eta}[Ga] \Bumpeq c \quad \textit{for all } a\ .$$

Proof Since G has prime inner face, by Proposition 8.15 we can express it as

$$G = \zeta(\mathsf{id}_X \otimes d_0 \otimes \cdots d_{k-1} \otimes D \otimes d_{k+1} \cdots \otimes d_{m-1})$$

for some $k \in m$ and some wiring $\zeta \colon X \uplus Y \to Z$, where $d_i \colon \langle Y_i \rangle$ $(i \neq k)$ and $D \colon 1 \to \langle Y_k \rangle$ are discrete and $Y = \biguplus_{i \in m} Y_i$.

Now any $a \colon \langle X \rangle$ can be expressed as $a = \omega d$ for some wiring $\omega \colon W \to X$ and discrete $d \colon \langle W \rangle$. Then we can express the composition Ga as follows:

$$Ga = \zeta(\omega \otimes \mathsf{id}_Y)(d_0 \otimes \cdots d_{k-1} \otimes (\mathsf{id}_W \otimes D)d \otimes d_{k+1} \cdots \otimes d_{m-1})\ .$$

Define $d_k \stackrel{\text{def}}{=} (\mathsf{id}_W \otimes D)d$; its names are $W \uplus Y_k$. Since d_k is discrete the expression for Ga is a DNF, and therefore by Definition 8.17 its instance by $\overline{\eta}$ is

$$\overline{\eta}[Ga] \bumpeq \zeta(\omega \otimes \mathsf{id}_Y)(d_{\eta(0)} \otimes \cdots \otimes d_{\eta(n-1)})\ .$$

Note that, since η is injective, the factors $d_{\eta(j)}$ have disjoint supports and name sets, so may be combined by $\otimes$ (rather than by $\parallel$ as in Definition 8.17). Since the instantiation may not be linear, we now distinguish two cases:

1. $\eta(k) = \ell$ for some $\ell \in n$. Then we may rewrite the instance as

$$\begin{aligned}
\overline{\eta}[Ga] &\bumpeq \zeta(\omega \otimes \mathsf{id}_Y)(d_{\eta(0)} \otimes \cdots \otimes d_\ell \otimes \cdots \otimes d_{\eta(n-1)}) \\
&= \zeta(\omega \otimes \mathsf{id}_Y)(d_{\eta(0)} \otimes \cdots \otimes (\mathsf{id}_W \otimes D)d \otimes \cdots \otimes d_{\eta(n-1)}) \\
&= \zeta(d_{\eta(0)} \otimes \cdots \otimes (\mathsf{id}_X \otimes D)\omega d \otimes \cdots \otimes d_{\eta(n-1)}) \\
&= Ca
\end{aligned}$$

where $C \stackrel{\text{def}}{=} \zeta(d_{\eta(0)} \otimes \cdots \otimes (\mathsf{id}_X \otimes D) \otimes \cdots \otimes d_{\eta(n-1)})$ is independent of a.

2. $\eta(k) \neq \ell$ for all $\ell \in n$. Then the inner names W of $\omega\colon W \to X$ are not among the names of $d_{\eta(0)} \otimes \cdots \otimes d_{\eta(n-1)}$. But it is easily seen that $\omega W \Bumpeq X$; hence

$$\begin{aligned}
\overline{\eta}[Ga] &\bumpeq \zeta(\omega \otimes \mathsf{id}_Y)(d_{\eta(0)} \otimes \cdots \otimes d_{\eta(n-1)}) \\
&= \zeta(\omega W \otimes \mathsf{id}_Y)(d_{\eta(0)} \otimes \cdots \otimes d_{\eta(n-1)}) \\
&\Bumpeq c
\end{aligned}$$

where $c \stackrel{\text{def}}{=} \zeta(X \otimes \mathsf{id}_Y)(d_{\eta(0)} \otimes \cdots \otimes d_{\eta(n-1)})$ is independent of a as required. ∎

We continue by considering the IPO underlying a standard transition $a \xrightarrow{L} a'$ with redex R. The IPO can be decomposed into an IPO pair, as shown in the diagram. We find that, for simple R, the diagram consists of pushouts. From now on we shall call a transition *simple* if its underlying redex is simple.

$$\begin{tikzcd}
\cdot \arrow[r, "L^{\mathsf{par}}"] & \cdot \arrow[r, "L^{\mathsf{red}}"] & \cdot \\
\cdot \arrow[u, "a"] \arrow[r, "d"'] & \cdot \arrow[u, "D^{\mathsf{par}}"'] \arrow[r, "\mathsf{id}_W \otimes R"'] & \cdot \arrow[u, "D"']
\end{tikzcd}$$

Proposition B.2 (transition pushouts) *In a concrete Brs, the IPO pair underlying a simple standard transition consists of pushouts.*

Proof Let the IPO pair underlying a transition $a \xrightarrow{L}_\lambda a'$ be as shown in the diagram. It will be enough to show that there can be no elisions in either IPO.

Since we are working in hard place graphs, there can be no place elisions. In the left square there can be no link elisions from d since, being discrete, it has no idle names; and there can be no link elisions from a into L^{par}, because the latter is open (since d is open). Thus the IPO is unique up to iso, hence a pushout. The argument for the right square is similar, using the simpleness of R. ∎

Next, we need two lemmas about non-engaged transitions.

Lemma B.3 *In a concrete Brs, suppose a simple standard transition is not engaged. Let its underlying IPO pair be as in the diagram. Then* $D^{\mathsf{par}} = D' \otimes \mathsf{id}_m$ *for some* D'*, up to isomorphism, where* m *is the inner face of* R*.*

Proof Since $|D^{\mathsf{par}}| \subseteq |a|$ we also have $|D^{\mathsf{par}}| \cap |R| = \emptyset$. Let K be the outer face of D^{par}. We have to prove, for each site $i \in m$, that i has no siblings in D^{par} and $D^{\mathsf{par}}(i) = k$ is a root in K.

Since R is guarding, $R(i) = v$ for some node v, hence $(L^{\mathsf{red}} D^{\mathsf{par}})(i) = v$. But v is not in D^{par} by assumption, so $D^{\mathsf{par}}(i) = k$ and $L^{\mathsf{red}}(k) = v$ for some root k. Now suppose i has a sibling, i.e. $D^{\mathsf{par}}(w) = k$ for some site or node $w \neq i$. Then we have $(L^{\mathsf{red}} D^{\mathsf{par}})(w) = v$, whence also $R(w) = v$. If w is a site this contradicts R inner-injective; if it is a node then it contradicts $|D^{\mathsf{par}}| \cap |R| = \emptyset$. Hence no such w can exist. This completes the proof. ∎

Lemma B.4 *In a concrete Brs let* a *be prime. Let* $a \xrightarrow{L}_{\lambda} a'$ *be a non-engaged simple standard transition based upon* (R, R', η)*, with underlying IPO pair as in the diagram. Let* $|a| \cap |d| \neq \emptyset$*. Then* $|a| \subseteq |d|$*, and* L^{red}*,* D *and* a' *take the following form up to iso:*

$$L^{\mathsf{red}} = \mathsf{id}_{W'} \otimes R\ , \quad D = \omega \otimes \mathsf{id}_J \quad \textit{and} \quad a' = (\mathsf{id}_{W'} \otimes R')\,\overline{\eta}[L^{\mathsf{par}} a]\ .$$

Proof From Lemma B.3 we find that D^{par} takes the form $D^{\mathsf{par}} = D' \otimes \mathsf{id}_m$ up to iso, where D' has domain W (with zero width) and m is the inner width of R.

We now claim that D' has no nodes. For there exists a node $u \in |a| \cap |d|$; if there exists any $v \in |D'|$ then also $v \in |a|$, hence (because a is prime) we would have u, v in the same region of $L^{\mathsf{par}} a$ but different regions of $D^{\mathsf{par}} d$, contradicting $L^{\mathsf{par}} a = D^{\mathsf{par}} d$. Thus $|a| \subseteq |d|$, and $D^{\mathsf{par}} = \omega \otimes \mathsf{id}_I$, with $\omega : W \to W'$ a wiring.

By Proposition B.2 the right-hand square in the diagram is a pushout, and hence a tensor IPO by Corollary 8.8. This yields the first two equations. For the third:

$$\begin{aligned} a' &= D(\mathsf{id}_W \otimes R')\,\overline{\eta}[d] \\ &= (\mathsf{id}_{W'} \otimes R')(\omega \otimes \mathsf{id}_{I'})\,\overline{\eta}[d] \\ (*) &= (\mathsf{id}_{W'} \otimes R')\,\eta[(\omega \otimes \mathsf{id}_I)d] \\ &= (\mathsf{id}_{W'} \otimes R')\,\overline{\eta}[L^{\mathsf{par}} a] \end{aligned}$$

where at $(*)$ we commute an instantiation with a wiring, by Proposition 8.18. ∎

We can now prove the adequacy theorem.

Theorem 9.11 (adequacy of engaged transitions) *In a simple prime affine Brs equipped with* ST*, the prime engaged transitions are adequate; that is, engaged bisimilarity $\sim^{\text{PE}}_{\text{ST}}$ coincides with bisimilarity $\sim_{\text{ST}}$ on prime agents.*

Proof We first treat the case of $\sim^{\text{PE}}_{\text{ST}}$ and $\sim_{\text{ST}}$, writing them as $\sim^{\text{PE}}$ and $\sim$ respectively.

It is immediate that $\sim\ \subseteq\ \sim^{\text{PE}}$ restricted to primes. For the converse we must prove that $a_0 \sim^{\text{PE}} a_1$ implies $a_0 \sim a_1$. An attempt to show that $\sim^{\text{PE}}$ is a standard bisimulation, i.e. a bisimulation for ST, does not succeed directly. Instead, we shall show that

$$\mathcal{S} = \{(Ca_0, Ca_1) \mid a_0 \sim^{\text{PE}} a_1\} \cup \Bumpeq$$

is a standard bisimulation up to support equivalence, relying on Proposition 4.5. This will suffice, for by taking $C = \mathsf{id}$ we deduce that $\sim^{\text{PE}}\ \subseteq\ \sim$.

Suppose that $a_0 \sim^{\text{PE}} a_1$. Let $Ca_0 \xrightarrow{M}_{\mu} b'_0$ be a standard transition, with MCa_1 defined. We must find b'_1 such that $Ca_1 \xrightarrow{M}_{\mu} b'_1$ and $(b'_0, b'_1) \in \mathcal{S}^{\bumpeq}$.

There exists a ground reaction rule (r_0, r'_0) and an IPO —the large square in diagram (a) below— underlying the given transition of Ca_0. Moreover E_0 is active, and if $\mathsf{width}(\mathsf{cod}(r_0)) = m$ then $\mathsf{width}(E_0)(m) = \mu$ and $b'_0 \bumpeq E_0 r'_0$. By taking an RPO for (a_0, r_0) relative to (MC, E_0) we get two IPOs as shown in the diagram.

Now D_0 is active, so the lower IPO underlies a transition $a_0 \xrightarrow{L}_{\lambda} a'_0 \stackrel{\text{def}}{=} D_0 r'_0$, where $\lambda = \mathsf{width}(D_0)(m_0)$. Also E is active at λ, and $b'_0 \bumpeq Ea'_0$. Since MCa_1 is defined we deduce that La_1 is defined, and we proceed to show in three separate cases the existence of a transition $a_1 \xrightarrow{L}_{\lambda} a'_1$, with underlying IPO as shown in diagram (b). (We cannot always infer such a transition for which $a'_0 \sim^{\text{PE}} a'_1$, even though we have $a_0 \sim^{\text{PE}} a_1$, since the transition of a_0 may not be engaged.) Substituting this IPO for the lower square in (a) then yields a transition

$$Ca_1 \xrightarrow{M}_{\mu} b'_1 \stackrel{\text{def}}{=} Ea'_1\ .$$

In each case we shall verify that $(b'_0, b'_1) \in \mathcal{S}^{\bumpeq}$, completing the proof of the theorem.

(a)
$$\begin{tikzcd} {} \arrow[r, "M"] & {} \\ {} \arrow[u, "C"] \arrow[r, "L"] & {} \arrow[u, "E"'] \\ {} \arrow[u, "a_0"] \arrow[r, "r_0"'] & {} \arrow[u, "D_0"'] \arrow[uu, bend right, "E_0"'] \end{tikzcd}$$

(b)
$$\begin{tikzcd} {} \arrow[r, "L"] & {} \\ {} \arrow[u, "a_1"] \arrow[r, "r_1"'] & {} \arrow[u, "D_1"'] \end{tikzcd}$$

Case 1 The transition of a_0 is engaged.
Then since r_0 is prime, by considering the IPO (L, D_0) and the outer face of D_0 we find that a'_0 is prime, so the transition lies in PE. So, since $a_0 \sim^{\text{PE}} a_1$, there exists a

transition $a_1 \xrightarrow{L}\!\!\triangleright_\lambda a_1'$ with $a_0' \sim^{\text{PE}} a_1'$. This readily yields the required transition of Ca_1.

Case 2 $|a_0| \cap |r_0| = \emptyset$.
Then the lower IPO of (a), being a pushout by Proposition B.2, is tensorial; so up to isomorphism we have

$$L = \mathsf{id}_H \otimes r_0 \ \text{ and } \ D_0 = a_0 \otimes \mathsf{id}\ .$$

Then $a_0' = E'a_0$, where $E' = \mathsf{id} \otimes r_0'$. Taking $C' \stackrel{\text{def}}{=} EE'$, we have $b_0' \simeq C'a_0$.

Form the IPO (b) by taking $r_1 = r_0$ and $D_1 = a_1 \otimes \mathsf{id}$; this underlies a transition $a_1 \xrightarrow{L}\!\!\triangleright_\lambda a_1' \stackrel{\text{def}}{=} E'a_1$. Substitute it for the lower square in (a), yielding a transition $Ca_1 \xrightarrow{M}\!\!\triangleright_\mu b_1' \stackrel{\text{def}}{=} Ea_1'$. Then $b_1' = C'a_1$, so $(b_0', b_1') \in \mathcal{S}^{\simeq}$ as required.

Case 3 The transition of a_0 is not engaged, but $|a_0| \cap |r_0| \neq \emptyset$.
Then there is a rule (R, R', η) with $|a_0| \cap |R| = \emptyset$, and a parameter d_0 such that

$$r_0 = (\mathsf{id}_{W_0} \otimes R)d_0 \ \text{ and } \ r_0' = (\mathsf{id}_{W_0} \otimes R')\,\overline{\eta}[d_0]\ .$$

Assume $R\colon m \to J$. Since a_0 is prime, from Lemma B.4 we find that, up to isomorphism, the IPO pair underlying the transition of a_0 takes the form of diagram (c) below, and moreover that $a_0' = (\mathsf{id}_{W'} \otimes R')\,\overline{\eta}[L^{\mathsf{par}}a_0]$.

(c)
$$\begin{tikzcd} {} \arrow[r, "L^{\mathsf{par}}"] & {} \arrow[r, "L^{\mathsf{red}} = \mathsf{id}_{W'}\otimes R"] & {} \\ {} \arrow[u, "a_0"] \arrow[r, "d_0"] & {} \arrow[u] \arrow[r, "\mathsf{id}_{W_0}\otimes R"] & {} \arrow[u, "\omega_0\otimes\mathsf{id}_J"'] \end{tikzcd}$$

(d)
$$\begin{tikzcd} {} \arrow[r, "L^{\mathsf{par}}"] & {} \arrow[r, "L^{\mathsf{red}} = \mathsf{id}_{W'}\otimes R"] & {} \\ {} \arrow[u, "a_1"] \arrow[r, "d_1"] & {} \arrow[u, "\omega_1\otimes\mathsf{id}_m"'] \arrow[r, "\mathsf{id}_{W_1}\otimes R"] & {} \arrow[u, "\omega_1\otimes\mathsf{id}_J"'] \end{tikzcd}$$

We shall now find a similar transition for a_1. We first consider $L^{\mathsf{par}}a_1$. Since d_0 is discrete we know by Proposition 8.16(1) that L^{par} is discrete; by Proposition 8.15 we can find a wiring $\omega_1\colon W_1 \to W'$ and discrete $d_1\colon W_1 \otimes m$ such that $L^{\mathsf{par}}a_1 = (\omega_1 \otimes \mathsf{id}_m)d_1$. By Proposition 8.16(2) this represents a pushout. By adjoining a tensorial pushout, we have an IPO pair as shown in diagram (d). Therefore by manipulations as in Lemma B.4 we have

$$\begin{aligned} a_1 \xrightarrow{L}\!\!\triangleright_\lambda a_1' &\stackrel{\text{def}}{=} (\omega_1 \otimes \mathsf{id}_J)(\mathsf{id}_{W_1} \otimes R')\,\overline{\eta}[d_1] \\ &= (\mathsf{id}_{W'} \otimes R')\,\overline{\eta}[L^{\mathsf{par}}a_1]\ . \end{aligned}$$

As in the previous case, this yields a transition $Ca_1 \xrightarrow{M}\!\!\triangleright_\mu b_1' \stackrel{\text{def}}{=} Ea_1'$. We now have

$$(b_0', b_1') = (F\,\overline{\eta}[L^{\mathsf{par}}a_0],\ F\,\overline{\eta}[L^{\mathsf{par}}a_1])$$

for a certain context F, where $a_0 \sim^{\text{PE}} a_1$ (both prime). Since $\overline{\eta}$ is affine, we can appeal to Proposition B.1 to find two cases. In the first case there is a context C

such that $\overline{\eta}[L^{\mathsf{par}} a] \bumpeq Ca$ for any a, and hence $(b_0', b_1') \in \mathcal{S}^{\bumpeq}$. In the second case there is a ground arrow c such that $\overline{\eta}[L^{\mathsf{par}} a] \Bumpeq c$ for any a, hence $b_0' \Bumpeq b_1'$, so $(b_0', b_1') \in \mathcal{S}$. Thus the bisimulation up to support equivalence is established.

This completes the proof of the theorem. ∎

As we have seen in case 1 of the proof, when a simple transition $a \xrightarrow{L}_{\lambda} a'$ is engaged, and a is prime, then so is a'. Thus, in proving the bisimilarity of prime agents, we can indeed confine attention to bisimulations containing only prime agents.