

An Inductive Characterization of Matching in Binding Bigraphs

Troels C. Damgaard¹, Arne J. Glenstrup¹, Lars Birkedal¹, and Robin Milner²

¹ IT University of Copenhagen, Rued Langgaards Vej 7, DK-2300 Copenhagen S, Denmark.

E-mail: {tcd,panic,birkedal}@itu.dk

² University of Cambridge, The Computer Laboratory, 15 JJ Thomson Avenue, Cambridge CB3 0FD, UK.

E-mail: Robin.Milner@cl.cam.ac.uk

Abstract. We analyze the matching problem for bigraphs. In particular, we present a sound and complete inductive characterization of matching in bigraphs with binding. Our results yield a specification for a provably correct matching algorithm, as needed by our prototype tool implementing bigraphical reactive systems.

Keywords: Bigraphs; bigraphical reactive systems; matching; complete inductive characterization

1. Introduction

Over the last decade, a theory of bigraphical reactive systems has been developed (JM04, Mil09, Mil05, Mil06). Bigraphical reactive systems (BRSs) provide a graphical model of computation in which both locality and connectivity are prominent. In essence, a *bigraph* consists of a *place graph*; a forest, whose nodes represent a variety of computational objects, and a *link graph*, which is a hyper graph connecting ports of the nodes. Bigraphs can be reconfigured by means of *reaction rules*. Loosely speaking, a *bigraphical reactive system* consists of a set of bigraphs and a set of reaction rules, which can be used to reconfigure the set of bigraphs. BRSs have been developed with principally two aims in mind: (1) to be able to model directly important aspects of ubiquitous systems by focusing on mobile connectivity and mobile locality, and (2) to provide a unification of existing theories by developing a general theory, in which many existing calculi for concurrency and mobility may be represented, with a uniform behavioural theory. The latter is achieved by representing the dynamics of bigraphs by an abstract definition of reaction rules from which a labelled transition system may be derived in such a way that an associated bisimulation relation is a congruence relation. (Recall that the notion of bisimulation is important since it expresses when two bigraphs are to be considered equal and that a relation is a congruence if it is closed under all bigraph contexts.) The unification has recovered existing behavioural theories for the π -calculus (JM04), the ambient calculus (Jen06), and has contributed to that for Petri nets (LM04). Thus the evaluation of the second aim has so far been encouraging. Birkedal et al. has begun to address the first aim, in particular, to show how to give bigraphical models of context-aware systems (BDE⁺06).

As suggested and argued in (JM04, Mil09, BDE⁺06, BBD⁺06) it would be very useful to have an implementation

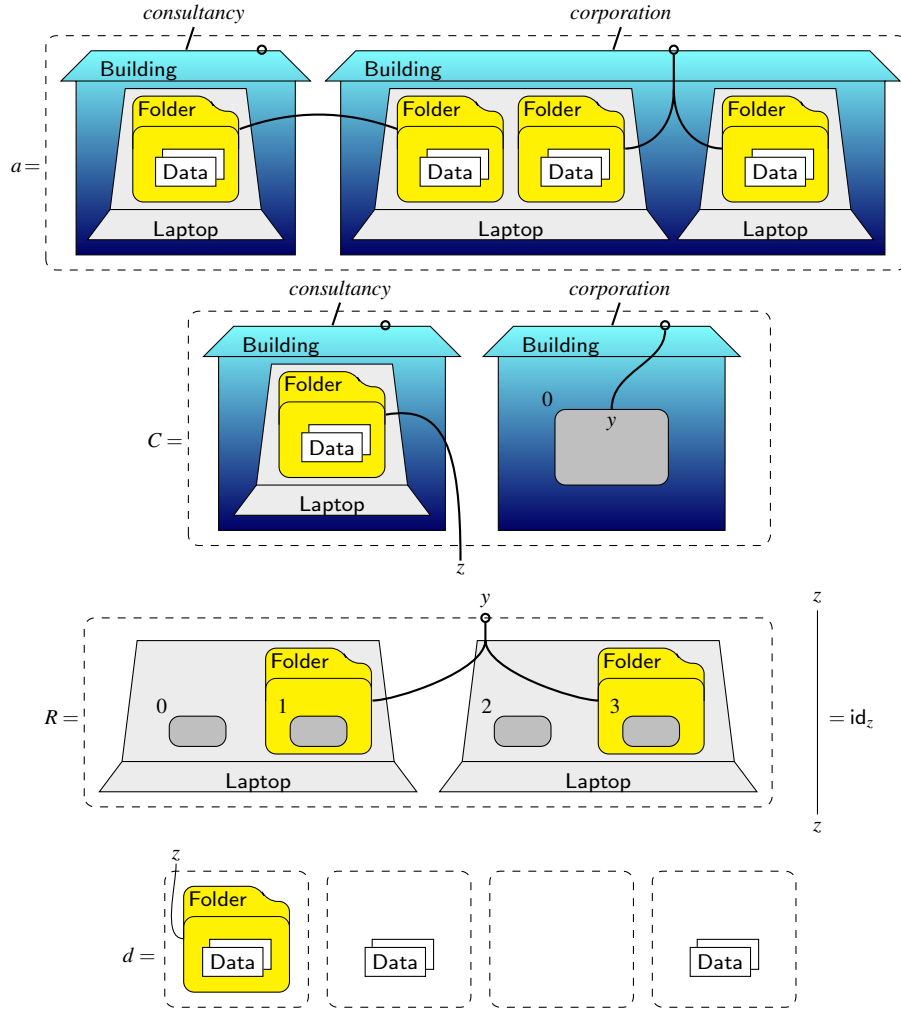


Figure 1. A bigraph $a = C \circ (R \otimes \text{id}_z) \circ d$ modeling data location and folder connectivity.

of the dynamics of bigraphical reactive systems to allow experimentation and simulation. In the Bigraphical Programming Languages research project at the IT University, we have been working towards such an implementation. The core problem of implementing the dynamics of bigraphical reactive systems is the *matching problem*, that is, to determine for a given bigraph and reaction rule whether and how the reaction rule can be applied to rewrite the bigraph. The topic of the present paper is to analyze the matching problem. We report on an implementation based on the work presented here elsewhere (GDHB10).

In Figure 1 we show several bigraphs. Consider the bigraph named a . It is intended to model two buildings, one belonging to a corporation and one belonging to a consultancy group. Inside the buildings are laptops with data nested inside folders. The nesting structure depicts the place graph. Links are used to name the buildings and, moreover, to model the association of folders to network channels. The laptop shown in the middle is intended to belong to a consultant working for the corporation—the consultant has a folder, containing some data, which is directly connected to a laptop in the consultancy (the link shown to the left) and a folder with a connection over the corporate backbone to another laptop in the corporation (the link shown to the right). There are two kinds of network channels in this example; those local to a building (i.e., over a building backbone) and global channels (presumably across the internet). The fact that folders are connected over the corporation backbone is expressed by linking those folders to a so-called binding

port on the corporation building, indicated by the circle. A binding port of a node imposes a scoping discipline to ensure that links connected to the port will be constrained to connections within the node.

Bigraphs can be rewritten by applying reaction rules—a reaction rule is a tuple $R \xrightarrow{\rho} R'$ specifying that the occurrence of a *redex* R can be replaced by a *reactum* R' . Bigraphs R and R' may contain holes, and ρ specifies how to fill each hole in R' with a copy of the contents of a hole in R . The abstract semantic definition of matching, as defined in the theory of bigraphs (Mil09), is roughly as follows (omitting many details): Given a reaction rule $R \xrightarrow{\rho} R'$ and a bigraph a , R occurs in a if we can find C, d, Z so that $a = C \circ (R \otimes \text{id}_Z) \circ d$; in which case it can be rewritten into $a' = C \circ (R' \otimes \text{id}_Z) \circ d'$, where d' is copied from d as specified by ρ . We call a the *agent*, C the *context* and d the *parameter*—they are all bigraphs. The operator \circ is vertical composition of bigraphs, while \otimes juxtaposes bigraphs horizontally. Z is a set of names exported by d . In short, we say that if the reaction rule *matches* the agent a , in the sense that a can be decomposed into a context C , redex R and a parameter d , then a can be rewritten into a' .

Consider the reaction rule $R \xrightarrow{\rho} R'$ with R from Figure 1, R' from Figure 2 and $\rho = \{0 \mapsto 0, 1 \mapsto 1, 2 \mapsto 2, 3 \mapsto 3, 4 \mapsto 1\}$. The intention of the reaction rule is to allow copying of data only between folders on two different but co-located laptops; we require also that the folders be connected over a network connection (note the link in R between the two folders), allowing that link to be possibly connected to a building backbone (expressed by linking the link to the so-called *local name* y). The agent a can be written as a composition of C , R and d —formally, $a = C \circ (R \otimes \text{id}_Z) \circ d$. Composition works by (1) plugging the roots (the dashed rectangles) of R and d into the holes (aka sites) of C respectively R ; (2) fusing together the connections between folder and z (in d) and z and folder (in C), removing the name z in the process; and (3) fusing together the connection between the local name y and the two folders in R and the name y and the bound port in C , removing the name y in the process. Note the use of id_z , a single link, in the composition $a = C \circ (R \otimes \text{id}_z) \circ d$; it allows a name z from the parameter d to be passed around the redex and be attached to something in the context C . The reactum R' contains a copy (as specified by ρ) of the site numbered 1 in R , expressing that data is copied between the shared folders. The sites numbered 0 and 2 in R allow the reaction rule to apply also when the laptops contain other folders than the two that are connected. Thus a can be rewritten using the reaction rule to another agent a' , shown in Figure 2, like a but with two data items in the rightmost laptop.

In the present paper we provide an *inductive characterization* of when there exists C , Z and d such that $a = C \circ (R \otimes \text{id}_Z) \circ d$ holds, by induction on the structure of a , R , C and d . It is a precise characterization in the sense that it is both sound and complete with respect to the abstract definition. This provides a detailed analysis of the matching problem, and give a specification for developing and *proving correct* an actual matching algorithm (which, given a and R , must find C , d , and Z such that $a = C \circ (R \otimes \text{id}_Z) \circ d$ holds). We further include a discussion of how one may derive matching algorithms directly from our inductive characterization.

Our inductive characterization is based on normal form theorems for binding bigraphs (DB06), which express how general bigraphs may be decomposed into a composition of simpler graphs. The normal form theorems and also the inductive characterization we present here is based on *discrete* decompositions of bigraphs, as specified by the grammar in Figure 4. This decomposition factors out global internal linking (like between the consultancy folder and corporation folder in Figure 1) in the ω of the G production, so that every point within the remaining discrete bigraph—the D production—is linked to exactly one global outer name, unless it is linked to a binding port. To a large extent, this simple linking allows us to analyze matching of a general bigraph by considering its link graph and place graph separately.

Importantly, by providing an abstract characterization founded in well-established theory for bigraphs, we expect to be able to combine or adapt more easily our approach to theory and techniques being developed for bigraphs; for instance, sortings (simple type disciplines) on bigraphs could be a source of early search elimination (BDH06).

The remainder of this paper is organized as follows: In Section 2 we give an informal description of binding bigraphs. The main contributions of this paper are in Section 3, where we present our inductive characterization of matching, and in the Appendix, where we give the proof of completeness of the characterization. To illustrate how the matching rules work together, in Section 4, we provide an inference tree for inferring the match in the example in Figure 1. Section 5 discusses how the inductive characterization yields a specification for a provably correct algorithm for matching. In the final sections we discuss related and future work, and conclude.

An extended abstract of this paper was presented at the GT-VC 2006 workshop (BDGM06). This extended and revised version fixes a few errors in the earlier presentations, provides more explanations and examples, and notably includes extensive details for the proof of completeness of the characterization and supporting lemmas, including a self-contained section on the algebraic properties of wirings and parallel product.

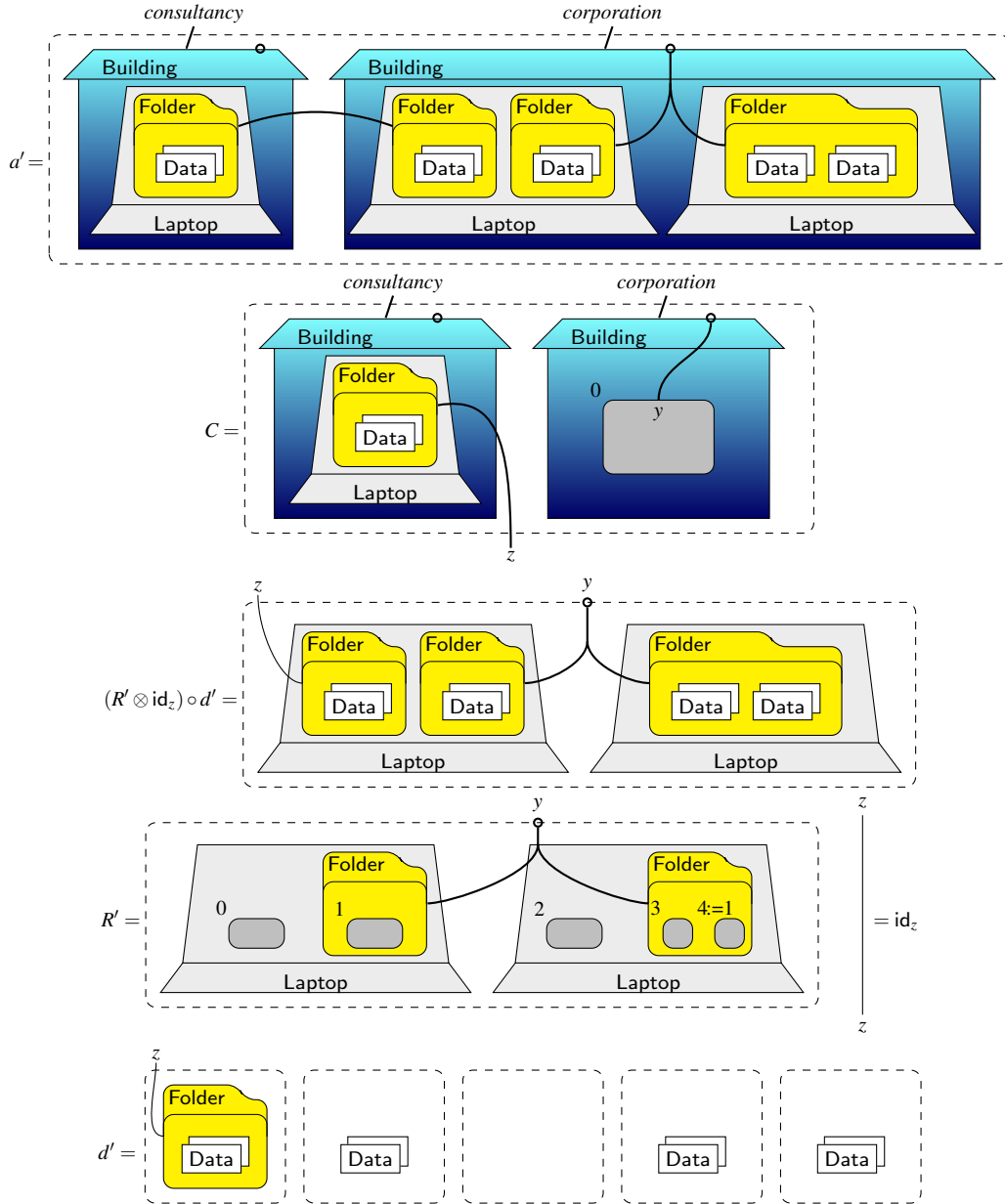


Figure 2. Applying reaction rule $R \xrightarrow{p} R'$ to bigraph a in Figure 1 copies data between connected folders, resulting in $d' = C \circ (R' \otimes id_z) \circ a'$.

2. Binding Bigraphs

In the following section, we present binding bigraphs fairly thoroughly, but we leave out formal details inessential for the present paper; for a more complete presentation, see (JM04, Mil09) or (DB06).

2.1. Concrete Bigraphs

A concrete binding bigraph G consists of a *place graph* G^P and a *link graph* G^L . The place graph is an ordered list of trees indicating *location*, with roots r_0, \dots, r_n , nodes v_0, \dots, v_k (some of which can be leaves), and a number of special leaves s_0, \dots, s_m called *sites*, while the link graph is a general graph over the node set v_0, \dots, v_k extended with

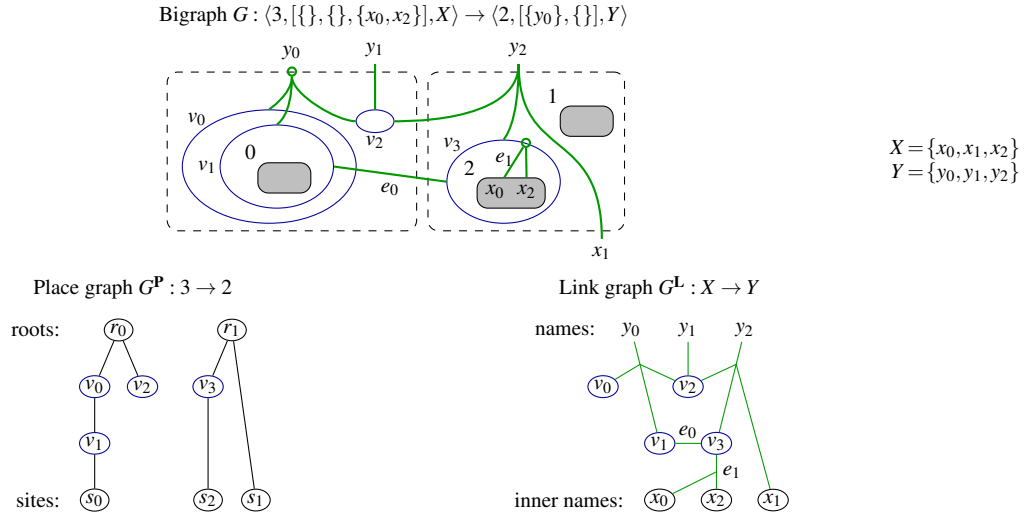


Figure 3. Example bigraph illustrated by nesting and as place and link graph.

inner names x_0, \dots, x_l , and equipped with hyperedges (i.e., edges that connect 0, 1, or more endpoints), indicating connectivity.

We usually illustrate the place graph by nesting nodes, as shown in the upper part of Figure 3 (ignore for now the interfaces denoted by “ $: \cdot \rightarrow \cdot$ ”). A *link* is a hyper edge of the link graph, either an internal *edge* e_0 or a *name* y . Those that are names are called *open*, internal edges are called *closed* links. Names and inner names can be *global* or *local*, the latter being located at a specific root or site, respectively. In Figure 3, y_0 is located at r_0 , indicated by a small ring, and x_0 and x_2 are located at s_2 , indicated by writing them within the site. Global names like y_1 and y_2 are drawn anywhere at the top, while global inner names like x_1 are drawn anywhere at the bottom. A link, including internal edges like e_1 in the figure, can be located with one *binder* (the ring), in which case it is a *bound link*, otherwise it is *free*. However, a bound link must satisfy the *scope rule*, a simple structural requirement that all points (see below) of the link lie within its location (in the place graph), except for the binder itself. This prevents y_2 and e_0 in the example from being bound.

2.2. Controls

Every node v has a *control* K indicated by $v : K$, which determines a binding and free arity $K : b \rightarrow f$. In the example of Figure 3, we could have $v_i : K_i, i = 0, 1, 2, 3$, where $K_0 : 0 \rightarrow 1$, $K_1 : 0 \rightarrow 2$, $K_2 : 0 \rightarrow 3$, $K_3 : 1 \rightarrow 2$. The arities determine the number of bound and free *ports* of the node, to which bound and free links, respectively, are connected. Ports and inner names are collectively referred to as *points*.

2.3. Abstract Bigraphs

While concrete bigraphs with named nodes and internal edges are the basis of bigraph theory (Mil09), our prime interest is in *abstract bigraphs*, equivalence classes of concrete bigraphs that differ only in the names of nodes and internal edges¹. Abstract bigraphs are illustrated with their node controls, as shown in Figure 1 with Building, Laptop, etc. In what follows, “bigraph” will thus mean “abstract bigraph.”

2.4. Interfaces

Every bigraph G has two *interfaces* I and J , written $G : I \rightarrow J$, where I is the *inner face* and J the *outer face*. An interface is a triple $\langle m, \vec{X}, X \rangle$, where m is the *width* (the number of sites or roots), X the entire set of local and global

¹ Formally, we also disregard *idle* edges: edges not connected to anything.

names, and \vec{X} are disjoint subsets of X indicating the locations of each local name, cf. Figure 3. We let $\varepsilon = \langle 0, [], \{\} \rangle$; when $m = 1$ the interface is *prime*, and if all $x \in X$ are located by \vec{X} , the interface is *local*. As in (Mil04) we write $G : \rightarrow J$ or $G : I \rightarrow$ for $G : I \rightarrow J$ when we are not concerned about I or J , respectively.

A bigraph $G : I \rightarrow J$ is called *ground*, or an *agent*, if $I = \varepsilon$, *prime* if I is local and J prime, and a *wiring* if $m = n = 0$, where m and n are the widths of I and J , respectively. For $I = \langle m, \vec{X}, X \rangle$, bigraph $\text{id}_I : I \rightarrow I$ consists of m roots, each root r_i containing just one site s_i , and a link graph linking each inner name $x \in X$ to name x .

2.5. Discrete and Regular Bigraphs

We say that a bigraph is *discrete* iff every free link is a name and has exactly one point. The virtue of discrete bigraphs is that any connectivity by internal edges must be bound, and node ports can be accessed individually by the names of the outer face. In Figure 1, only R, R' and d are discrete, because the free internal edges of a and C have two points. Further, a bigraph is *name-discrete* iff it is discrete and every bound link is either an edge, or (if it is a name in the outer face) has exactly one point. Note that name-discrete implies discrete.

A bigraph is *regular* if, for all nodes v and sites i, j, k with $i \leq j \leq k$, if i and k are descendants of v , then j is also a descendant of v . Further, for roots $r_{i'}$ and $r_{j'}$, and all sites i and j where i is a descendant of $r_{i'}$ and j of $r_{j'}$, if $i \leq j$ then $i' \leq j'$. The bigraphs in the figures are all regular, the permutation in Table 1 is not. The virtue of regular bigraphs is that permutations can be avoided when composing them from basic bigraphs.

2.6. Tensor Product, Parallel Product, and Composition

For bigraphs G_1 and G_2 that share no names or inner names, we can make the *tensor product* $G_1 \otimes G_2$ by juxtaposing their place graphs, constructing the union of their link graphs, and increasing the indexes of sites in G_2 by the number of sites of G_1 . For instance, bigraph d of Figure 1 is a tensor product of four primes. We write $\bigotimes_i^n G_i$ for the iterated tensor $G_0 \otimes \cdots \otimes G_{n-1}$, which, in case $n = 0$, is id_ε .

The *parallel product* $G_1 \parallel G_2$ is like the tensor product, except global names can be shared: if y is shared, all points of y in G_1 and G_2 become the points of y in $G_1 \parallel G_2$.

We can *compose* bigraphs $G_2 : I \rightarrow I'$ and $G_1 : I' \rightarrow J$, yielding bigraph $G_1 \circ G_2 : I \rightarrow J$, by plugging the sites of G_1 with the roots of G_2 , eliminating both, and connecting names of G_2 with inner names of G_1 —as in Figure 1, where $a = C \circ (\text{id}_\varepsilon \otimes R) \circ d$. In the following, we will omit the ‘ \circ ’, and simply write $G_1 G_2$ for composition, letting it bind tighter than tensor product.

2.7. Active, Passive and Atomic Controls

In addition to arity, each control is assigned a *kind*, either atomic, active or passive, and describe nodes according to their control kinds. We require that atomic nodes contain no nodes except sites; any site being a descendant of a passive node is *passive*, otherwise it is *active*. If all sites of a bigraph G are active, G is *active*.

2.8. Bigraphical Reactive Systems

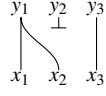
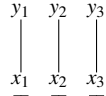
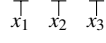
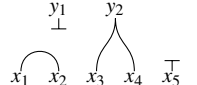
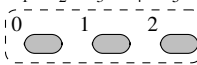
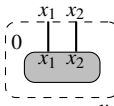
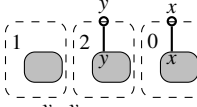
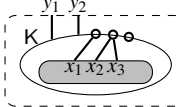
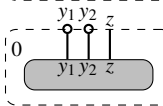
Bigraphs in themselves model two essential parts of context: locality and connectivity. To model also *dynamics*, we introduce *bigraphical reactive systems* (BRS) as a set of *rules*. Each rule $R \rightarrow_\rho R'$ consists of a regular *redex* $R : I \rightarrow J$, a *reactum* $R' : I' \rightarrow J'$, and an *instantiation* ρ , mapping each site of R' to a site of R^2 . Interfaces $I = \langle m, \vec{X}, X \rangle$ and $I' = \langle m', \vec{X}', X' \rangle$ must be local, and are essentially related by $X'_i = X_{\rho(i)}$.³ We illustrate ρ by a ‘ $i := j$ ’, as shown in Figure 1, whenever $\rho(i) = j \neq i$. Given an instantiation ρ and a discrete bigraph $d = d_0 \otimes \cdots \otimes d_k$ with prime d_i ’s, we let $\rho(d) = d_{\rho(0)} \otimes \cdots \otimes d_{\rho(k)}$, allowing copying, discarding and reordering parts of d .

Given an agent a , a *match* of redex R is a decomposition $a = C(\text{id}_Z \otimes R)d$, with active context C , discrete parameter

² The direction of ρ from R' to R allows copying from one site of R to 0, 1 or more sites of R' .

³ When copying sites with local names, one has to also rename local names of copied sites, which makes the relation slightly less straightforward. We elide the details here, as they are not central to matching, and refer the reader to (JM04, Chapter 12) for the full formal details.

Table 1. Basic bigraphs, abstraction, and metavariables ranging over bigraphs.

	Notation	Example
Substitution σ	$\vec{y}/\vec{X} : X \rightarrow Y$	$[y_1, y_2, y_3]/[\{x_1, x_2\}, \{\}, \{x_3\}] =$ 
Renaming α, β	$\vec{y}/\vec{x} : X \rightarrow Y$	$[y_1, y_2, y_3]/[x_1, x_2, x_3] =$ 
Closure	$/X : X \rightarrow \{\}$	$/\{x_1, x_2, x_3\} =$ 
Wiring ω	$\omega : X \rightarrow Y$	$([y_1, y_2]/[y_1, y_2] \otimes [\{z_1, z_2\}]) / [y_1, z_1, y_2, z_2] / [\{\}, \{x_1, x_2\}, \{x_3, x_4\}, \{x_5\}] =$ 
Merge	$merge_n : n \rightarrow 1$	$merge_3 =$ 
Concretion	$\ulcorner X \urcorner : (X) \rightarrow \langle X \rangle$	$\ulcorner \{x_1, x_2\} \urcorner =$ 
Permutation $\pi_{\vec{X}}, \pi$	$\{i \mapsto j, \dots\}_{\vec{X}} : (\vec{X}) \rightarrow (\pi(\vec{X}))$	$\{0 \mapsto 2, 1 \mapsto 0, 2 \mapsto 1\}_{[\{x\}, \emptyset, \{y\}]} =$ 
Ion	$K_{y(\vec{X})} : (\langle \vec{X} \rangle) \rightarrow \langle \{\vec{y}\} \rangle$	$K_{[y_1, y_2]}([\{x_1\}, \{x_2, x_3\}, \{\}]) =$ 
Abstraction	$(Y)P : I \rightarrow \langle 1, [Y], Z \uplus Y \rangle$	$(\{y_1, y_2\}) \ulcorner \{y_1, y_2, z\} \urcorner =$ 

d and its global names Z . Dynamics is achieved by transforming a into a new agent $a' = C(\text{id}_Z \otimes R')d'$, where $d' = \rho(d)$ —an example is shown in Figure 1. This definition of a match is as in (Mil09), except that we here also require R to be regular. The restriction to regular redexes R , which simplifies the inductive characterization, does not limit the set of possible reactions, because sites in R and R' can be renumbered to render R regular.

2.9. Notation, Basic Bigraphs, and Abstraction

In the sequel, we will use the following notation: \uplus denotes union of sets required to be disjoint⁴; we write $\{\vec{Y}\}$ for $Y_0 \uplus \dots \uplus Y_{n-1}$ when $\vec{Y} = Y_0, \dots, Y_{n-1}$, and similarly $\{\vec{y}\}$ for $\{y_0, \dots, y_{n-1}\}$. For interfaces, we write n to mean $\langle n, [\emptyset, \dots, \emptyset], \emptyset \rangle$, X to mean $\langle 0, [], X \rangle$, $\langle X \rangle$ to mean $\langle 1, [\{\}], X \rangle$, (X) to mean $\langle 1, [X], X \rangle$, and (\vec{X}) to mean $\langle n, \vec{X}, \{\vec{X}\} \rangle$, when the length of \vec{X} is n .

Any bigraph can be constructed by applying composition, tensor product and *abstraction* to identities (on all interfaces) and a minimal set of basic bigraphs (DB06). Given a prime P , the abstraction operation localizes a subset of its outer names. The scope rule is necessarily respected since the inner face of a prime P is required to be local, so all points of P are located within its root. The abstraction operator is denoted by (\cdot) and reaches as far right as possible.

We illustrate the basic bigraphs and abstraction in Table 1. Substitutions introduce open linking and closures create edges. Renamings are one-one substitutions, while wirings ω range over all expressions built by composition and tensor from substitutions and closures. A *merge* bigraph merges sites into a single root, a concretion maps local inner names to global outer names, while a permutation serves to produce any ordering of sites inside roots. The ion introduces nodes, mapping free ports to global outer names and bound ports to sets of local inner names.

⁴ Thus, premises, side conditions and other expressions involving $A \uplus B$ can only be valid and true if $A \cap B = \emptyset$.

$M ::= (K_{\vec{y}(\vec{X})} \otimes \text{id}_Z)(X)P$	<i>molecule</i>
$Q ::= \lceil \alpha \rceil \mid M$	<i>singular top-level node</i>
$P ::= (\text{merge}_n \otimes \text{id}_Y)(\otimes_{i \in n} Q_i)$	<i>global discrete prime</i>
$N ::= (\widehat{\sigma} \otimes \text{id}_Z)(X)P$	<i>discrete prime</i>
$D ::= (\otimes_{i \in n} N_i)\pi$	<i>discrete bigraph without global inner names</i>
$G ::= (\omega \otimes \text{id}_{(X)})(D \otimes \text{id}_Y)$	<i>binding bigraph</i>

Figure 4. Grammar for bigraph G on normal form; for regular bigraphs, $\pi = \text{id}$.

The resulting inductive language for building bigraphs is fairly heavy, but it is easy to derive more sugared languages, resembling closely standard notation used for mobile process calculi, based on these basic bigraphs and operators. In this paper, we shall not be concerned much with concrete notation, though; we refer the interested reader to (BDE⁺06) or the forthcoming tutorial book on bigraphs by Milner.

We shall only use a few basic conventions for shortening expressions. For a renaming $\alpha : X \rightarrow Y$, we write $\lceil \alpha \rceil$ to mean $(\alpha \otimes \text{id}_1) \lceil X \rceil$, and when $\sigma : U \rightarrow Y$, we let $\widehat{\sigma} = (Y)(\sigma \otimes \text{id}_1) \lceil U \rceil$. We write substitutions $\vec{y}/[\emptyset, \dots, \emptyset] : \varepsilon \rightarrow Y$ as Y . For permutations, when used in any context, $\pi_{\vec{X}}G$ or $G\pi_{\vec{X}}$, \vec{X} is given entirely by the interface of G ; hence, we shall typically elide the names of $\pi_{\vec{X}}$ and write only π . Note that $[\]/\] = / \emptyset = \pi_0 = \text{id}_\varepsilon$ and $\text{merge}_1 = \lceil \emptyset \rceil = \pi_1 = \text{id}_1$, where π_i is the nameless permutation of width i .

To conclude this section, we illustrate the basic bigraphs and operations by showing expressions for some of the bigraphs in the previous examples. The bigraph of Figure 3 can be expressed as

$$\begin{aligned}
G &= (\omega \otimes ((\{y_0\})(y_0/Y_0 \otimes \text{id}_1) \lceil Y_0 \rceil))(((Y_0)P_1) \otimes P_2 \otimes y_2/x_1), \text{ where} \\
\omega &= (/e_0 \otimes \text{id}_{\{y_1, y_2\}})[y_1, y_2, e_0]/[\{y_1\}, \{y_2, y_2', y_2''\}, \{e_0, e_0'\}], \quad Y_0 = \{y_0, y_0', y_0''\} \\
P_1 &= (\text{id}_{\{y_0, y_1, y_2', e_0\}} \otimes \text{merge}_2) \left((\text{id}_{\{y_0, e_0\}} \otimes K_0[y_0']) K_1[y_0, e_0] \otimes K_2[y_0', y_1, y_2'] \text{merge}_0 \right) \\
P_2 &= (\text{id}_{\{e_0', y_2''\}} \otimes \text{merge}_2) (K_3[e_0', y_2''](\{x_0, x_2\}) \otimes \lceil \emptyset \rceil),
\end{aligned}$$

and for Figure 1 we have $a = (\text{id}_{\{\text{consultancy}, \text{corporation}\}} \otimes /z)(p_1 \parallel p_2)$, where

$$\begin{aligned}
p_1 &= (\text{id}_z \otimes \text{Building}_{[\text{consultancy}]}(\{\{\}\}) \text{Laptop}) \text{Folder}_{[z]} \text{Data merge}_0 \\
p_2 &= (\text{id}_z \otimes \text{Building}_{[\text{corporation}]}(\{\{y_1, y_2\}\}))(\{y_1, y_2\})(\text{id}_{\{z, y_1, y_2\}} \otimes \text{merge}_2)(p_2' \otimes p_2'') \\
p_2' &= (\text{id}_{\{z, y_1\}} \otimes \text{Laptop merge}_2)(\text{Folder}_{[z]} \text{Data merge}_0 \otimes \text{Folder}_{[y_1]} \text{Data merge}_0) \\
p_2'' &= (\text{id}_{y_2} \otimes \text{Laptop}) \text{Folder}_{[y_2]} \text{Data merge}_0
\end{aligned}$$

3. Inductive Characterization of Matching

In this section we present our inductive characterization of matching. To ease the presentation we shall disregard the requirement that the context in a match must be active (it is straightforward to extend the presentation to include that requirement). To simplify notation we shall write id for local identity bigraphs, without a subscript showing the interface, when it is clear from the context what interface is intended. Furthermore, we use the name *molecule* for a prime with just one outermost node.

3.1. Discrete decomposition

We base our characterization on discrete decomposition of bigraphs, as shown in Figure 4, which separates global (or free) wiring from the place graph and local wiring. The following proposition expresses how any bigraph may be decomposed into a global wiring ω , and a discrete bigraph D (cf. Section 2.5).

Proposition 3.1 (Discrete decomposition). Any bigraph G can be decomposed into a composition of the following form

$$G = (\omega \otimes \text{id})(D \otimes \text{id}_Y),$$

where D is discrete and with local inner face. Any other decomposition of G on this form takes the form $G = (\omega' \otimes \text{id})(D' \otimes \text{id}_Y)$, where $\omega' = \omega(\alpha \otimes \text{id}_Y)$ and $D' = (\alpha^{-1} \otimes \text{id})D$, for suitable α .

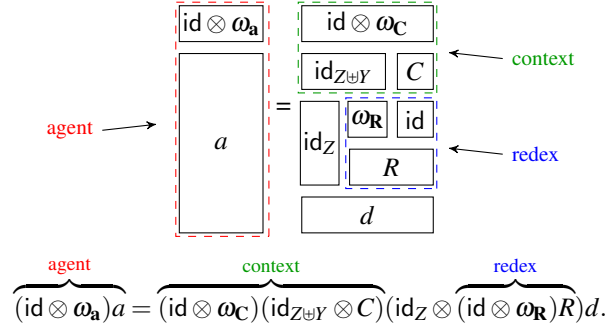


Figure 5. Illustrating valid matching sentences

This is proven by factoring out the top-level local linking of D , and factoring id_Y into D , resulting in the expression $(\omega \otimes \hat{\sigma})D'$, and then applying the normal form theorem of (DB06).

3.2. Matching Sentences

We now define matching sentences and rules for deriving valid matching sentences. Given general a', C', R', d for which we want to infer a match $a' = C'(R' \otimes \text{id}_Z)d$, then by Proposition 3.1, we can decompose d', C' , and R' , obtaining $a' = (\text{id} \otimes \omega_a)a$, $C' = (\text{id} \otimes \omega_C)(C \otimes \text{id}_Y \otimes \text{id}_Z)$, and $R' = (\text{id} \otimes \omega_R)R$. The matching sentences now relate the constituents of these decompositions:

Definition 3.2 (Matching sentence). A *matching sentence* is a 7-place relation among wirings and bigraphs, written $\omega_a, \omega_R, \omega_C \vdash a, R \hookrightarrow C, d$, where $\omega_a, \omega_R, \omega_C$ are wirings, and a, R, C, d are discrete bigraphs, R and C have local inner faces, and R is regular.

Definition 3.3 (Valid matching sentence). A matching sentence $\omega_a, \omega_R, \omega_C \vdash a, R \hookrightarrow C, d$, where $\omega_R : \rightarrow Y$, and d has global outer names Z , is *valid*, denoted $\omega_a, \omega_R, \omega_C \vDash a, R \hookrightarrow C, d$, iff

$$(\text{id} \otimes \omega_a)a = (\text{id} \otimes \omega_C)(C \otimes \text{id}_Y \otimes \text{id}_Z)(\text{id}_Z \otimes (\text{id} \otimes \omega_R)R)d.$$

where unqualified identities are local and determined from their context.

Note that for a valid sentence $\omega_a, \omega_R, \omega_C \vdash a, R \hookrightarrow C, d$, if we let $a' = (\text{id} \otimes \omega_a)a$, $C' = (\text{id} \otimes \omega_C)(C \otimes \text{id}_Y \otimes \text{id}_Z)$, and $R' = (\text{id} \otimes \omega_R)R$, then $a' = C'(R' \otimes \text{id}_Z)d$. Thus, valid sentences precisely capture the abstract definition of matching.

Rearranging a few identities, we can illustrate the discrete decomposition of the agent, context, and redex in a valid matching sentence schematically as in Figure 5. We draw bigraph composition as vertical composition, and tensor product as horizontal juxtaposition.

3.3. Rules for Matching

In Figure 6 and Figure 7, we present a set of rules and axioms for inferring matching sentences. In PAR we require further that the tensor products of all discrete components be defined. Also, in the premises of the rules PERM and ION, and in the conclusion of rules MERGE, ION, and SWITCH we require the id 's to have width 0 (hence be link graph identities). This determines them entirely from the context.

Valid matching sentences can be inferred using the rules by following the structure of the bigraph normal forms in Figure 4. We now explain each of the rules, and to illustrate how the matching rules work together, in the following section (Section 4) we provide an inference tree for inferring the match in the example depicted in Figure 1.

Given a permutation π and n primes, we can find a *pushed-through* permutation $\bar{\pi}$ depending only on π and the inner faces of the primes, such that $\pi \circ \otimes_i^n P_i = (\otimes_i^n P_{\pi^{-1}(i)}) \circ \bar{\pi}$ (DB06, Lemma 2). Using this, the PERM rule simply pushes a permutation on the inside of the context through the redex, permuting the discrete primes, and producing a pushed-through permutation $\bar{\pi}$, depending on π and the inner face of the redex.

The PAR rule explains how to match a product, given two valid matches. The two valid matches are allowed to *share*

$$\begin{array}{c}
\text{PERM} \frac{\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \vdash a, \bigotimes_i^m P_{\pi^{-1}(i)} \hookrightarrow C, (\bar{\pi} \otimes \text{id})d}{\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \vdash a, \bigotimes_i^m P_i \hookrightarrow C\pi, d} \\
\\
\text{PAR} \frac{\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \parallel \omega \vdash a, R \hookrightarrow C, d \quad \omega_{\mathbf{b}}, \omega_{\mathbf{S}}, \omega_{\mathbf{D}} \parallel \omega \vdash b, S \hookrightarrow D, e}{\omega_{\mathbf{a}} \parallel \omega_{\mathbf{b}}, \omega_{\mathbf{R}} \parallel \omega_{\mathbf{S}}, \omega_{\mathbf{C}} \parallel \omega_{\mathbf{D}} \parallel \omega \vdash a \otimes b, R \otimes S \hookrightarrow C \otimes D, d \otimes e} \\
\\
\text{LSUB} \frac{\sigma_{\mathbf{a}} \otimes \omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \sigma_{\mathbf{C}} \otimes \omega_{\mathbf{C}} \vdash p, R \hookrightarrow P, d}{\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \vdash (\widehat{\sigma}_{\mathbf{a}} \otimes \text{id})(Z)p, R \hookrightarrow (\widehat{\sigma}_{\mathbf{C}} \otimes \text{id})(U)P, d}, \sigma_{\mathbf{a}} : Z \rightarrow U, \sigma_{\mathbf{C}} : U \rightarrow \\
\\
\text{MERGE} \frac{\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \vdash a, R \hookrightarrow C, d}{\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \vdash (\text{merge} \otimes \text{id})a, R \hookrightarrow (\text{merge} \otimes \text{id})C, d}, a \text{ global} \\
\\
\text{ION} \frac{\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \vdash ((\vec{v})/(\vec{X}) \otimes \text{id})p, R \hookrightarrow ((\vec{v})/(\vec{Z}) \otimes \text{id})P, d}{\sigma \parallel \omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \sigma \alpha \parallel \omega_{\mathbf{C}} \vdash (K_{\vec{y}(\vec{X})} \otimes \text{id})p, R \hookrightarrow (K_{\vec{u}(\vec{Z})} \otimes \text{id})P, d}, \alpha = \vec{y}/\vec{u}, \sigma : \{\vec{y}\} \rightarrow \\
\\
\text{SWITCH} \frac{\omega_{\mathbf{a}}, \text{id}_{\varepsilon}, \omega_{\mathbf{C}}(\alpha \sigma \otimes \omega_{\mathbf{R}} \otimes \text{id}_Z) \vdash p, \text{id} \hookrightarrow P, d}{\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \vdash p, (\widehat{\sigma} \otimes \text{id})(W)P \hookrightarrow \ulcorner \alpha \urcorner, d}, \sigma : W \rightarrow U, d : \langle m, \vec{X}, X \uplus Z \rangle \\
\\
\text{CLOSE} \frac{\sigma_{\mathbf{a}}, \sigma_{\mathbf{R}}, \text{id}_Y \otimes \sigma_{\mathbf{C}} \vdash a, R \hookrightarrow C, d}{(\text{id} \otimes / (Y \uplus X))\sigma_{\mathbf{a}}, (\text{id} \otimes / Y)\sigma_{\mathbf{R}}, (\text{id} \otimes / X)\sigma_{\mathbf{C}} \vdash a, R \hookrightarrow C, d}, \sigma_{\mathbf{C}} : \rightarrow Z \uplus X, \sigma_{\mathbf{R}} : \rightarrow U \uplus Y
\end{array}$$

Figure 6. Rules for matching binding bigraphs

$$\begin{array}{c}
\text{PRIME-AXIOM} \frac{\alpha : X \rightarrow U \quad \beta : V \rightarrow Z \quad \sigma : U \uplus Z \rightarrow \quad \tau : Y \rightarrow X \quad p : \langle Y \uplus Z \rangle}{\sigma(\alpha \tau \otimes \beta), \text{id}_{\varepsilon}, \sigma \vdash p, \text{id}_{(X)} \hookrightarrow \ulcorner \alpha \urcorner, (X)(\tau \otimes \beta \otimes \text{id}_1)p} \\
\\
\text{WIRING-AXIOM} \frac{}{y, Y, y/Y \vdash \text{id}_{\varepsilon}, \text{id}_{\varepsilon} \hookrightarrow \text{id}_{\varepsilon}, \text{id}_{\varepsilon}}
\end{array}$$

Figure 7. Axioms for matching binding bigraphs

some context wiring ω , if the redices share (global) names. Figure 8 illustrates the conclusion of a match using PAR. The wirings are depicted above the underlying discrete bigraphs: a product of two agents a and b containing a single node; a product of two contexts C and D , which contain only a site; and, a product of two redices R and S containing a single node (for this example, the parameters are empty). As the parallel product of the redex wirings $\omega_{\mathbf{R}}$ and $\omega_{\mathbf{S}}$ already maps the links from w_1 and w_2 to a shared name w , the link from w is shared wiring ω (while the links from x and y_1 are in $\omega_{\mathbf{a}} = \omega \parallel \omega_{\mathbf{C}}$, and the links from y_2 and z are in $\omega_{\mathbf{b}} = \omega \parallel \omega_{\mathbf{D}}$).

The LSUB rule allows us to match any discrete prime by matching an underlying *free* prime with the wiring of agent and context extended with the underlying global substitutions $\sigma_{\mathbf{a}}$ and $\sigma_{\mathbf{C}}$. In other words, this rule expresses that we can match a single-rooted bigraph with *local* names by matching the corresponding free bigraph (i.e., forgetting the locality of the names).

The MERGE rule simply states that if we can match (global) bigraphs with several roots, then we can *merge* those roots into a single root, and still have a valid match.

The ION rule states intuitively that if we have two valid matches with primes in agent and context, we can compose both primes with an ion (i.e., a node with wiring) and still have a valid match. For any given match of discrete primes,

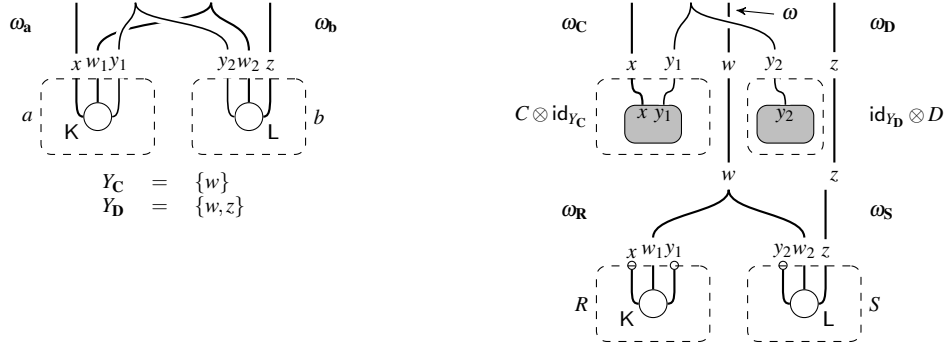
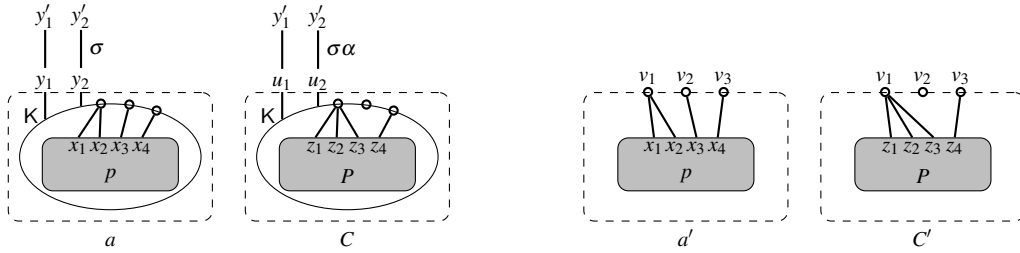


Figure 8. Matching a product using the PAR rule


 Figure 9. Matching an ion in the agent a with context C by matching a' with context C'

we can compose with ions $K_{\vec{y}(\vec{X})}$ or $K_{\vec{u}(\vec{Z})}$, if we extend the wirings of agents and contexts with isomorphic wiring on the outer names \vec{y} and \vec{u} ; stated in the rule by requiring that we extend with σ and $\sigma\alpha$ (where $\alpha = \vec{y}/\vec{u}$). For example, if we seek to match the agent $a = (\text{id} \otimes K_{\vec{y}(\vec{X})})p$ with a context $C = (\text{id} \otimes K_{\vec{u}(\vec{Z})})P$, then it suffices to consider matching of $a' = (\vec{v})/(\vec{X})p$ with a context $C' = (\vec{v})/(\vec{Z})P$, as illustrated in Figure 9. The local linkage remaining in a' and C' is the local substitutions underlying the ions in a and C .

Given an agent and considering an inference tree operationally bottom up, the rules specify how to decompose the agent while *constructing* the corresponding context (cf., e.g., the ION rule). At the point where the root of the redex is matched, the SWITCH rule is applied, switching the redex into context position, so that further decomposition of the agent *checks* that the redex matches. A redex root needs to be matched when the only remainder of the context is a site, possibly with some local linkage. Thus, when inferring a match, every rule except SWITCH can be used in two modes: one where the agent and redex are given, resulting in a context and parameter; and, after a SWITCH, one where the agent and context are given, resulting in a parameter. This is reflected in the fact, that in the premise of SWITCH, in the matching sentence the redex-position is id and the redex-wiring is empty, since we are now only concerned with checking the redex, and constructing the parameter. (Note, that this means that for $d : \langle 1, (Y), Y \uplus Z \rangle$, the unspecified id in the context-wiring is necessarily id_Z .)

In Figure 10, we depict an application of SWITCH. As illustrated the agent does not change when applying the rule; we simply try to infer a match between the current agent structure a , and a new context C' constructed as the composition of the former context C composed with the former redex R . As required, the new redex R' is an identity and the parameter d is preserved.

The PRIME-AXIOM and WIRING-AXIOM axioms are the base cases of matching inferences. The latter axiom is used to match bigraphs of zero width (i.e., bigraphs which only contain wiring). By iterated application of this axiom and PAR, we handle matching of any idle names (i.e., outer names not connected to anything). Intuitively, this axiom allows us to only be concerned with wiring with no idle names (i.e., epi wiring) in all other rules (cf., Note A.21 in the appendix, immediately following the proof of Lemma 3.8 concerned with WIRING-AXIOM.)

The PRIME-AXIOM handles the case where we have matched all nodes in the redex and context; this is the case when only sites remain. The axiom expresses this for primes only; as PAR allows us to combine several valid matches with product, most other rules simply need to be concerned with the prime case. Hence, the axiom requires both redex and context to be single sites (in the context allowing renaming of local names), and requires the agent and parameter place graph to be equal and their wiring to be compatible.

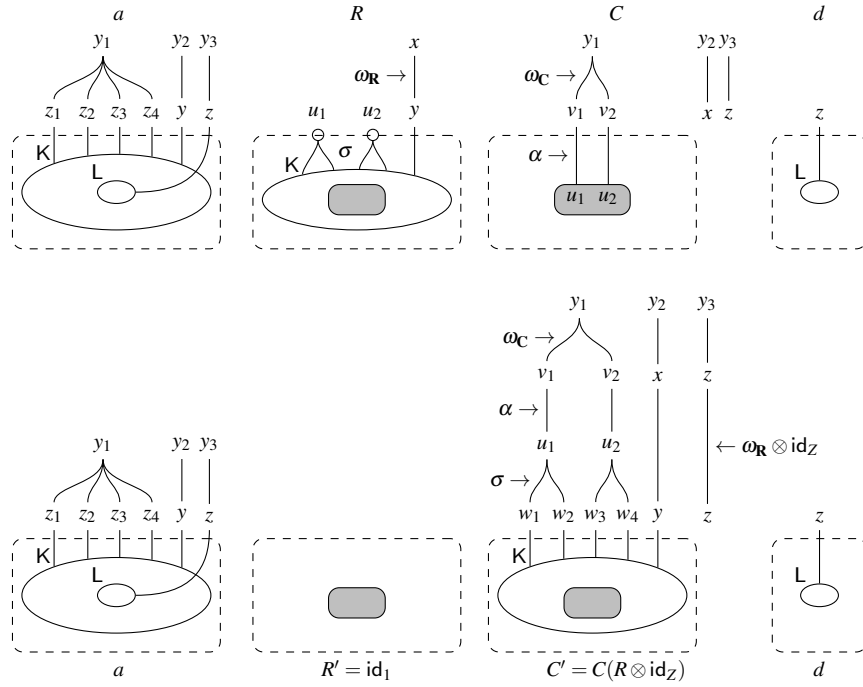


Figure 10. Matching $a = C(R \otimes \text{id}_Z)d$ by matching $a = C'(R' \otimes \text{id}_Z)d$ using the SWITCH rule.

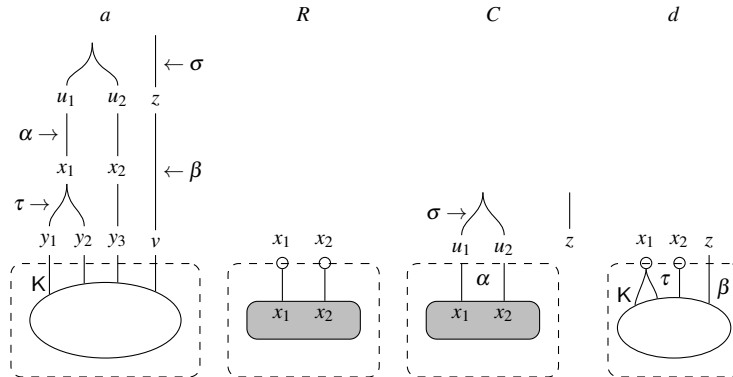


Figure 11. Matching $a = C(R \otimes \text{id}_Z)d$ by using PRIME-AXIOM.

Figure 11 depicts an instance of PRIME-AXIOM. The agent and parameter contain a node of the same control K , and the wirings τ , β , α , and σ , which are composed to determine the wiring of $C(R \otimes \text{id}_Z)d$, matches the wiring of a (which in the figure is split accordingly, to illustrate this match).

Finally, the CLOSE rule allows us to infer a match for bigraphs where all global links are open, and “close” this match by replacing names in wirings with edges, cf. Figure 12.

3.4. Soundness and Completeness of the Characterization

The two following theorems state that the rules constitute a sound and complete characterization of matching.

Theorem 3.4 (Soundness). The rules for matching in Figures 6 and 7 are sound, that is, any derivable matching sentence is valid.

Proof. Straightforward, by standard algebraic manipulations. \square

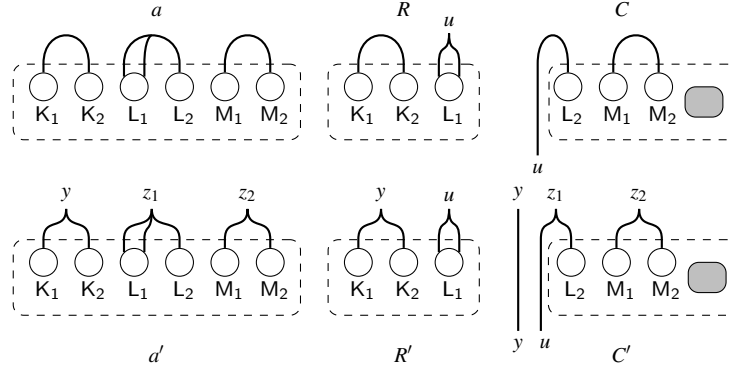


Figure 12. Matching closed links within and between redex and context

The completeness theorem is proved by induction on the size of valid sentences, which is defined as follows.

Definition 3.5 (Size of a matching sentence). The size of a matching sentence $\omega_a, \omega_R, \omega_C \vdash a, R \hookrightarrow C, d$ is the number of ions in a .

The following lemmas express how a valid sentence may be derived by applications of inference rules to valid sentences of lesser or equal size. The proofs proceed by first decomposing the components of the given valid sentence, then defining the components of the valid sentence(s) claimed to exist and, finally, verifying that (1) the sentences claimed to exist really are valid and (2) that the given sentence can indeed be derived as claimed. The decompositions are obtained via Proposition 3.1 and other normal forms for binding bigraphs, and the verifications proceed using uniqueness results for normal forms based on those found in (DB06). We give extensive details for the proofs of these lemmas in the Appendix.

Lemma 3.6. Every valid sentence $\omega_a, \omega_R, \omega_C \vdash a, R \hookrightarrow C, d$ is provable using the CLOSE and the PERM rules on a valid sentence, of equal size, of the form $\sigma_a, \sigma_R, \sigma_C \vdash a, S \hookrightarrow \bigotimes_i^n P_i, e$.

Lemma 3.7. Every valid sentence $\sigma_a, \sigma_R, \sigma_C \vdash a, R \hookrightarrow \bigotimes_i^n P_i, d$, with each P_i prime and discrete, is provable using the PAR rule on valid sentences, of lesser or equal size, of the form $\sigma_a^0, \sigma_R^0, \sigma_C^0 \parallel \sigma_C^S \vdash p, S \hookrightarrow P_0, e$ and $\sigma_a^1, \sigma_R^1, \sigma_C^1 \parallel \sigma_C^S \vdash d', R' \hookrightarrow \bigotimes_{i=1}^n P_i, e'$. All substitutions mentioned above are required to be epi (i.e., with no idle names).

Lemma 3.8. Every valid sentence $\sigma_a, \sigma_R, \sigma_C \vdash a, R \hookrightarrow \text{id}_e, d$ is provable using PAR and WIRING-AXIOM.

Lemma 3.9. Every valid sentence $\sigma_a, \sigma_R, \sigma_C \vdash p, R \hookrightarrow P, d$, with p and P prime and discrete, is provable using the LSUB rule on a valid sentence, of lesser or equal size, of the form $\sigma'_a, \sigma'_R, \sigma'_C \vdash p', R \hookrightarrow P', d$, where p' and P' are discrete free primes. All substitutions mentioned above are required to be epi (i.e., with no idle names).

Lemma 3.10. Every valid sentence $\sigma_a, \sigma_R, \sigma_C \vdash p, R \hookrightarrow Q, d$, with p and Q discrete and free primes, is provable using the MERGE, PAR (iterated), and SWITCH rules on valid sentences, each of lesser or equal size, and each on one of two forms:

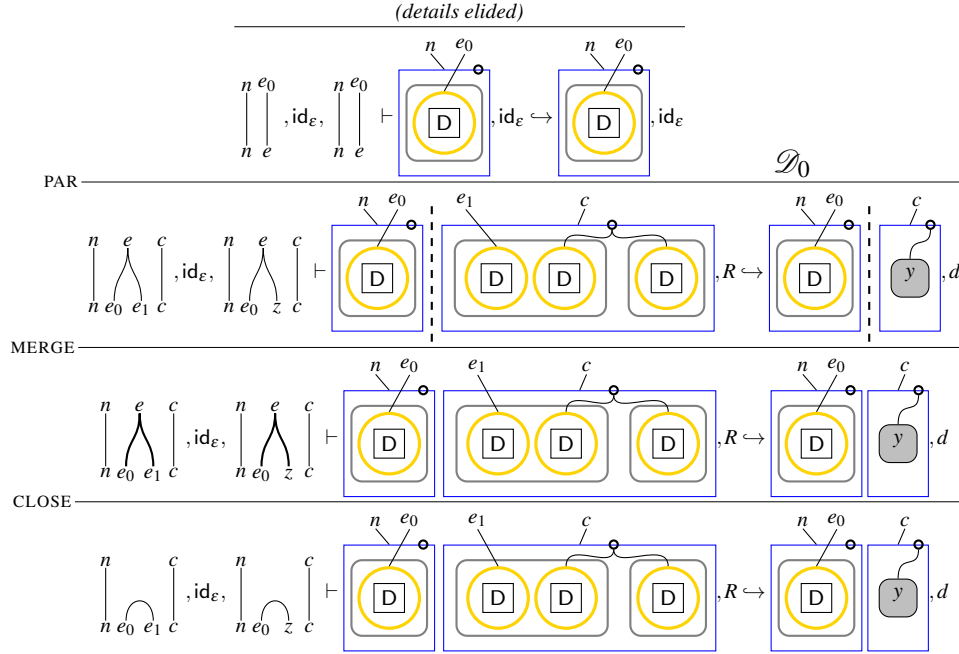
- $\sigma'_a, \sigma'_R, \sigma'_C \vdash p^N, \text{id} \hookrightarrow P^N, e$, where p^N and P^N are free discrete primes,
- $\sigma'_a, \sigma'_R, \sigma'_C \vdash m, S \hookrightarrow M, e$, where m and M are free discrete molecules.

All substitutions mentioned above are required to be epi (i.e., with no idle names).

Lemma 3.11. Every valid sentence $\sigma_a, \sigma_R, \sigma_C \vdash m, R \hookrightarrow M, d$, with m and M free discrete molecules, is provable using the ION rule on a valid sentence $\sigma'_a, \sigma'_R, \sigma'_C \vdash p, R \hookrightarrow P, d$, of lesser size, where p and P are discrete primes. All substitutions mentioned above are required to be epi (i.e., with no idle names).

Lemma 3.12. Every valid sentence $\sigma_a, \sigma_R, \sigma_C \vdash p, \text{id} \hookrightarrow P, e$, with p and P free discrete primes, is provable using the MERGE and PAR (iterated) rules on valid sentences of equal or lesser size, which are either instances of rule PRIME-AXIOM or of the form $\sigma'_a, \sigma'_R, \sigma'_M \vdash m, R \hookrightarrow M, d$. All substitutions mentioned above are required to be epi (i.e., with no idle names).

Theorem 3.13 (Completeness). The rules for matching in Figures 6 and 7 are complete, that is, any valid matching sentence can be derived from the rules.



Proof. By induction on the size of a sentence. By the lemmas above, we have that all valid sentences with size n can be derived from valid sentences of the form $\sigma_a, \sigma_R, \sigma_C \models m, R \leftrightarrow M, d$, with m and M free discrete molecules, of size less than or equal to n . By Lemma 3.11, these can be derived from sentences of size less than n . \square

4. An Example: Inferring a Match

In this section, we give an inference tree for inferring the match in the example depicted in Figure 1. To fit the inference tree in three reasonably small figures (Figures 13, 14, and 15), we use a more humble visual style, than in Figure 1, to depict roots, nodes and names.

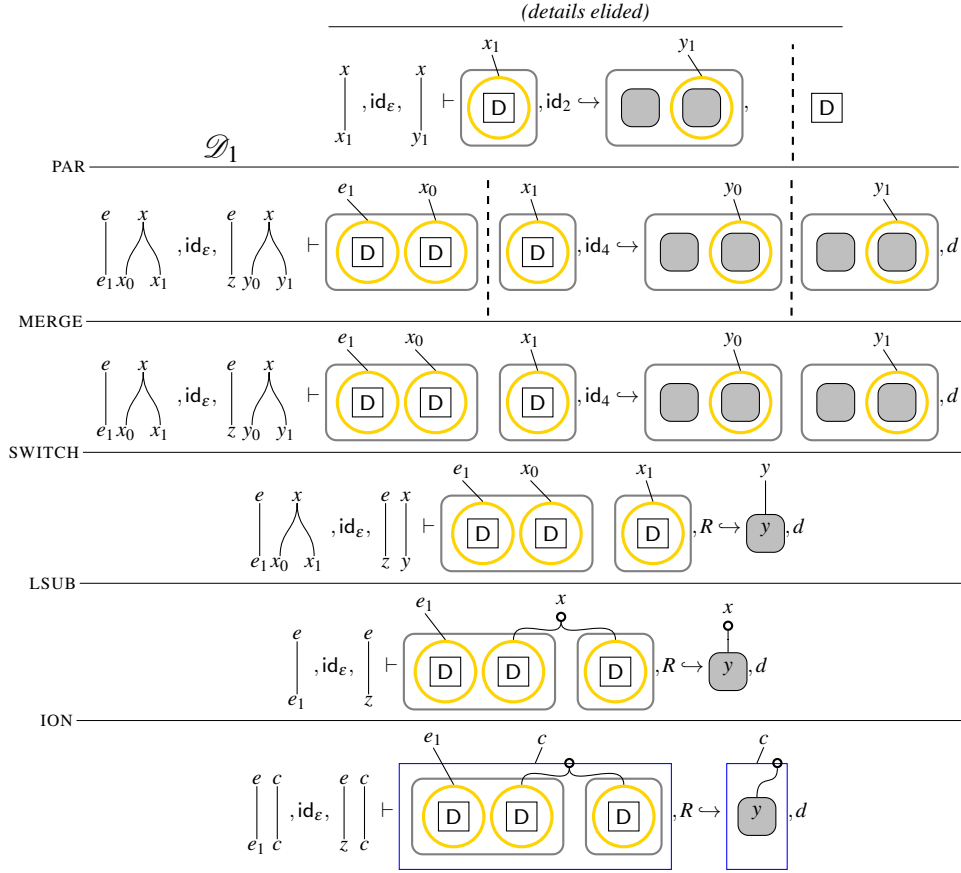
Roots are only drawn when there are more than one; in that case we use a dashed separating line to indicate separate roots (see for example the conclusion of PAR in Figure 13). Controls of nodes are indicated with the shape (and colour) of the node: Buildings are (blue) rectangles, laptops are (gray) rectangles with rounded corners, folders are (yellow) circles, and data-nodes are black squares with a D inside. Instead of the name *consultancy* we use n and instead of *corporation* we use c . Finally, we do not depict the basic redex R and parameter d , which are illustrated already in Figure 1.

We build the inference bottom up and start by decomposing a and C discretely to obtain a sentence that we aim to prove as the conclusion of an application of CLOSE in Figure 13. (Note that in contrast to a and C , bigraphs R and d are already discrete as depicted in Figure 1.) The application of CLOSE allows us to match and introduce the edge between names e_0 and e_1 in the agent, and between names e_0 and z in the context. We are building an inference bottom up, so in the premise we simply introduce a fresh outer name e to map these names to.

Next, we aim to use an application of PAR to pair up two inferences of matches between top-level nodes of the agent and the context. The top-level nodes of the agent and the context are in the same root, though, so after using PAR to pair up two such inferences, we need to apply MERGE to merge the roots introduced by PAR into a single root.

Matching the left root of the agent to the left root of the context is a simple matter, as the agent and context are isomorphic, both in wiring and underlying discrete bigraph, while the redex and parameter are empty (consequently, we leave out the details for that subderivation). The derivation \mathcal{D}_0 is depicted in Figure 14.

In the conclusion of the derivation \mathcal{D}_0 we use an application of ION to match the top-level building node of the right root of the agent to the building node of the right root of the context. Reading the application of the ION rule bottom up, we see that removing the building node in both agent and context requires us to match and remove the


 Figure 14. The derivation \mathcal{D}_0

global wiring upon free ports of the building node (e/e_1 and e/z , respectively), and introduce a common local outer name x to map linkage upon binding ports to.

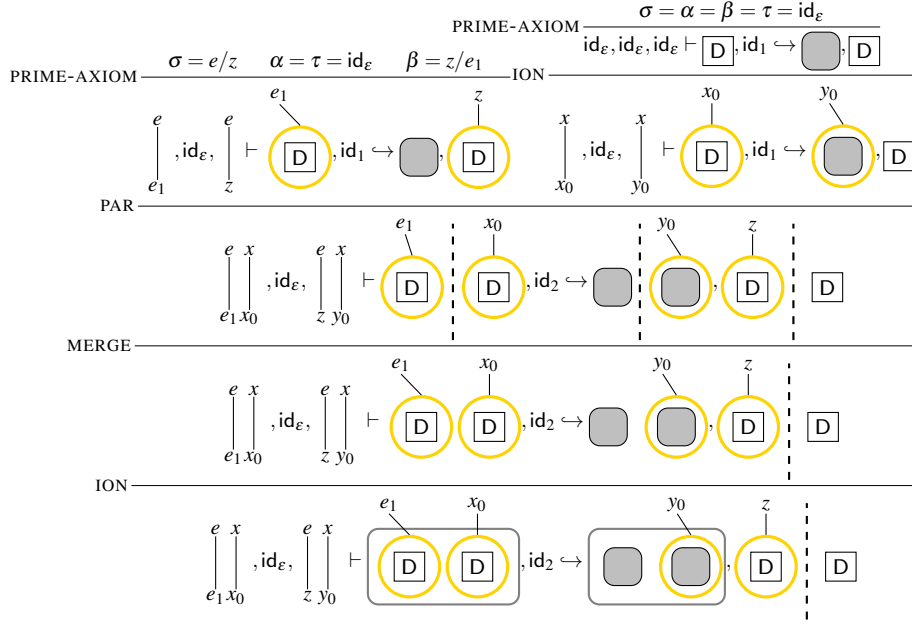
We immediately proceed to remove that local name. The LSUB rule allows to introduce local names and local wiring; so when building up an inference tree, we simply introduce new global names (in this case, x_0 and x_1 in the agent, and y in the context), and wire those names through global wiring instead.

In the resulting premise of LSUB the only remainder of the context is a concretion $\lceil y \rceil$ (i.e., a single site), so we can SWITCH to matching the redex. This means that we need to try to infer a match between the current agent structure (in the derivation) and the remainder of the context (i.e., the context wiring and $\lceil y \rceil$) composed with the redex R —with identity redex and the same parameter d . The composition $\lceil y \rceil R$ makes global the name y of R , which is subsequently wired to an x in the context-wiring. Consequentially, in the context in the premise of SWITCH we have R with two global names y_0 and y_1 , which are wired through the link $x/\{y_0, y_1\}$. The redex in the premise is id_4 to match the four roots of the parameter d .

Now, we aim again to use an application of PAR to pair up two inferences of matches between the top-level nodes of the current agent and context. After using PAR to pair up those inferences, we need again to apply MERGE to merge the roots introduced by PAR into a single root. The remaining two derivations for matching folders with data in the agent to folders and data in the (former) redex and parameter are very similar; we show the leftmost derivation \mathcal{D}_1 in Figure 15.

We conclude the derivation \mathcal{D}_1 as we did \mathcal{D}_0 , namely with an application of ION, in this case, to match and remove a laptop node. Folder nodes have no ports, so here we need not be concerned with wiring.

Another application of the rules MERGE and PAR allows to combine two derivations, the left of which is an instance of PRIME-AXIOM. The PRIME-AXIOM requires both redex and context to be single sites (in the context allowing renaming of local names), and requires the agent and parameter place graph to be equal and their wiring to be compatible. In this case, only the names e_1 and z differ, but as they are internal (i.e., disappear when composing the wiring

Figure 15. The derivation \mathcal{D}_1

with the discrete underlying bigraph), we can construct suitable wirings $\sigma = e/z$ and $\beta = z/e_1$ to verify that we have a valid instance of PRIME-AXIOM.

In the remaining rightmost derivation we need a single application of ION, before being able to conclude the entire derivation with a very simple instance of PRIME-AXIOM, as data nodes have no ports to connect wiring to.

5. Towards Algorithms for Matching

The completeness theorem tells us that we can find all valid matching sentences by applications of the rules for matching. Thus the rules for matching define an algorithm for matching, for instance easily expressed in Prolog, which simply operates by searching for inference trees using the rules.

Although we can base a matching algorithm directly upon the matching rules, we do not claim that an efficient matching algorithm has to be so based. We have introduced matching rules for a dual purpose: first, to characterize matching structurally and inductively in order to understand it (in particular, to understand the relation to representations based on normal forms and to understand exactly where choices between different matches can be made during matching); second, to provide a specification from which to begin the search for truly efficient matching algorithms, and to verify them against. This rigorous approach to matching is justified, in our view, because matching will be the workhorse of any implementation of bigraph dynamics.

In practice, one is, of course, interested in minimizing unnecessary blind search, and thus, for instance, only search for inference trees of a certain form. Indeed, one can show that it suffices to consider so-called *normal inference trees*, which put restrictions on the order in which the inference rules are applied (such as, e.g., always concluding with the CLOSE rule). We shall not include a formal definition of normal inference trees here, but rather discuss some of the possibilities for defining normal inference trees. We first remark that to retain completeness, any definition of normal inference must, of course, ensure no loss of provability. Looking at the formulations of the lemmas leading up to the completeness theorem, we see that there are indeed several possibilities for the definition of normal inference tree. For example, from Lemma 3.6 we see that we are free to conclude each inference tree with CLOSE and then PERM or vice versa. Further, in several rules we are allowed to propagate closed links, even though CLOSE intuitively makes that unnecessary. We have chosen to leave this freedom in the rule system and instead comment on how we could *extend* the set of rules to allow even more freedom in choosing our definition of normal inference tree. This is important when thinking about implementations, as each definition of normal inference tree corresponds to a different algorithmic approach to matching.

One may say that the current set of rules naturally give rise to normal inferences that are a mix between matching

the link graph “lazily” or “eagerly”. Instead of the CLOSE rule, one could have amended the PAR and ION rules (those with \parallel in the conclusion) such that they would also handle matching of closures. This would have allowed true “by need” link-matching. Conversely, one could have amended the CLOSE rule to also compare substitutions, allowing us to consider matching of discrete bigraphs up to renamings (i.e., isomorphisms) on their outerfaces. If we amended the LSUB and SWITCH rules to work accordingly, this would actually preclude the need for the wirings $\omega_a, \omega_R, \omega_C$ in matching sentences. It seems, though, that the tedious complexity added into these rules would mean that we would gain little in removing complexity from the rules as a whole. Anyhow, these changes would allow us to define a variant of normal inferences, which would be “strict” in the link graph, in that we would immediately be able to reject possible matches based on the link graph (instead of the place graph).

Another possibility would be to add a rule GLOB, allowing us to match all wiring stemming from a *single* prime as global wiring. This idea seems to indicate that matching in *local* bigraphs (Mil04) (where there is no global linkage but instead multilocal names) could be handled similarly, by recasting the rules to work on local links and just locating names at all roots where they occur.

An implementation of matching must, of course, represent bigraphs in some way. One possibility is to represent bigraphs directly by place and link graphs, and then implement the normal form lemmas, which express how bigraphs may be decomposed into simpler bigraphs; then matching can proceed by induction on the decomposed graph. In general, however, the “decomposition functions” return *sets* of possible decompositions, because normal forms are only unique up to certain permutations. (For example, $merge(M_1 \otimes M_2) = merge(M_2 \otimes M_1)$.) A matching implementation needs to explore all the possible decompositions. This can be made explicit formally, by phrasing the inductive characterization of matching not on bigraphs but on *expressions* (i.e., syntax) for binding bigraphs. Doing so forces us to add an inference rule, which allows one to replace any expression in a matching sentence $\omega_a, \omega_R, \omega_C \vdash a, R \hookrightarrow C, d$, say a , by a' , when a' is provably equal to a via the axioms for structural equality of bigraphs (DB06). Doing so clearly yields a complete set of rules on bigraphical expressions, but yields a wildly nondeterministic inference system as we might need to apply equality axioms between every step to infer a match. (This is reminiscent of problems arising when implementing rewriting logic, i.e., term rewriting modulo a set of static equivalences.) Consequentially, normal inference trees for rules based on syntax, needs to spell out *how* and *where* to apply equality axioms. The definition of normal inference trees will then *formally explicate* all the possibilities that a matching algorithm needs to explore.

Based on the considerations above, we have worked out a definition of normal inference tree for matching bigraph expressions and proved it complete. In particular, we also utilize normal forms for expressions by defining normal inferences that require each inference to start by rewriting the term to be on normal form. This restricts considerably the set of expressions that the normal inferences need be concerned with.

This definition of normal inferences is the basis of a prototype implementation of a tool for working with bigraphical reactive systems. We report on this in another paper (GDHB10). The prototype tool is also available online at:

<http://tiger.itu.dk:8080/bplweb/>

6. Related and Future Work

Bigraphical reactive systems are related to general graph transformation systems; see (EEPT06) for a recent comprehensive overview of graph transformation systems. In particular, bigraph matching is strongly related to the general graph pattern matching (GPM) problem, so general GPM algorithms might be applicable (Ull76, Fu97, LV02, Zün94). Due to the special structure of bigraphs, general GPM algorithms are expected to be inefficient, although some GPM tools (VVF05) use heuristic search strategies that might be able to discover and exploit bigraph structure.

A special aspect of bigraphs is that we may match a set of subtrees with a single node (site) in the redex, and match multiple redex roots in different places within the agent. Fu (Fu97) handles such wildcard nodes and multiple patterns, but directly applying his algorithm is not straightforward, as he attacks the problem of tree isomorphism of rooted graphs unfolded to finite unbounded depths. The subtree isomorphism problem (Sel77, Val02, ST99) is simpler than GPM, but applying it directly to the place graphs of bigraphs would not exploit the constraints imposed by the link graphs. Rather, efficient implementations of bigraph matching should be derived from the initial implementation by experimenting with different normal inference tree definitions, and combining it with subtree isomorphism algorithms. The inductive characterization provided here will make it easier to prove an actual algorithm correct. Finally, as noted in the introduction, by providing an abstract characterization founded in well-established theory for bigraphs, we expect to be able to combine or adapt more easily our approach to theory and techniques being developed for bigraphs.

For a more detailed account of related work, in particular on relations between BRSs, graph transformations, term rewriting and term graph rewriting, see (Dam06, Section 6).

Future work on bigraph matching include investigating how we may combine, for instance, our approach with sortings on bigraphs (BDH06), which could be a source of early search elimination. We are also considering rephrasing the rules to derive a set of constraints for wirings (the three first components of a matching sentence), which could be fed to a constraint solving algorithm, instead of matching them online as the rules.

7. Conclusion

We have presented a sound and complete inductive characterization of matching for binding bigraphs. The characterization provides a formal specification for a matching algorithm for binding bigraphs; and, even further, the specification has already served as the basis for an implementation of a prototype tool for working with BRSs.

References

- [BBD⁺06] Lars Birkedal, Mikkel Bundgaard, Troels Christoffer Damgaard, Søren Debois, Ebbe Elsborg, Arne John Glenstrup, Thomas Troels Hildebrandt, Robin Milner, and Henning Niss. Bigraphical programming languages for pervasive computing. In Thomas Strang, Vinny Cahill, and Aaron Quigley, editors, *Proceedings of Pervasive 2006 International Workshop on Combining Theory and Systems Building in Pervasive Computing*, pages 653–658, May 2006.
- [BDE⁺06] Lars Birkedal, Søren Debois, Ebbe Elsborg, Thomas Troels Hildebrandt, and Henning Niss. Bigraphical models of context-aware systems. In Luca Aceto and Anna Ingólfssdóttir, editors, *Proceedings of the 9th International Conference on Foundations of Software Science and Computation Structure*, volume 3921 of *Lecture Notes in Computer Science*, pages 187–201. Springer-Verlag, March 2006.
- [BDGM06] Lars Birkedal, Troels C. Damgaard, Arne John Glenstrup, and Robin Milner. Matching of bigraphs. In *Proceedings of Graph Transformation for Verification and Concurrency Workshop 2006*, Electronic Notes in Theoretical Computer Science. Elsevier, August 2006.
- [BDH06] Lars Birkedal, Søren Debois, and Thomas Troels Hildebrandt. Sortings for reactive systems. In Christel Baier and Holger Hermanns, editors, *Proceedings of the 17th International Conference on Concurrency Theory*, volume 4137 of *Lecture Notes in Computer Science*, pages 248–262. Springer-Verlag, August 2006.
- [Dam06] Troels C. Damgaard. Syntactic theory for bigraphs. Master’s thesis, IT University of Copenhagen, December 2006.
- [DB06] Troels C. Damgaard and Lars Birkedal. Axiomatizing binding bigraphs. *Nordic Journal of Computing*, 13(1-2):58–77, 2006.
- [EEPT06] Hartmut Ehrig, Karsten Ehrig, Ulrike Prange, and Gabriele Taentzer. *Fundamentals of Algebraic Graph Transformation*. Monographs in Theoretical Computer Science. An EATCS Series. Springer, 2006.
- [Fu97] James Jianghai Fu. Directed graph pattern matching and topological embedding. *Journal of Algorithms*, 22(2):372–391, 1997.
- [GDHB10] Arne John Glenstrup, Troels Christoffer Damgaard, Espen Højsgaard, and Lars Birkedal. An implementation of bigraph matching. Technical Report TR-2010-135, IT University of Copenhagen, December 2010.
- [Jen06] Ole H. Jensen. Mobile processes in bigraphs, 2006. Monograph available at <http://www.cl.cam.ac.uk/~rm135/Jensen-monograph.html>.
- [JM04] Ole H. Jensen and Robin Milner. Bigraphs and mobile processes (revised). Technical Report UCAM-CL-TR-580, University of Cambridge, February 2004.
- [LM04] James J. Leifer and Robin Milner. Transition systems, link graphs and Petri nets. Technical Report UCAM-CL-TR-598, University of Cambridge, August 2004.
- [LV02] Javier Larrosa and Gabriel Valiente. Constraint satisfaction algorithms for graph pattern matching. *Mathematical Structures in Computer Science*, 12:403–422, 2002.
- [Mil04] Robin Milner. Bigraphs whose names have multiple locality. Technical Report UCAM-CL-TR-603, University of Cambridge, September 2004.

- [Mil05] Robin Milner. Axioms for bigraphical structure. *Mathematical Structures in Computer Science*, 15(6):1005–1032, 2005.
- [Mil06] Robin Milner. Pure bigraphs: structure and dynamics. *Information and Computation*, 204(1):60–122, 2006.
- [Mil09] Robin Milner. *The Space and Motion of Communicating Agents*. Cambridge University Press, Cambridge, 2009.
- [Sel77] Stanley M. Selkow. The tree-to-tree editing problem. *Information Processing Letters*, 6(6):184–186, 1977.
- [ST99] Ron Shamir and Dekel Tsur. Faster subtree isomorphism. *Journal of Algorithms*, 33(2):267–280, 1999.
- [Ull76] Jeffrey D. Ullman. An algorithm for subgraph isomorphism. *Journal of the ACM*, 23(1):31–42, 1976.
- [Val02] Gabriel Valiente. *Algorithms on Trees and Graphs*. Springer-Verlag, Berlin, 2002.
- [VVF05] Gergely Varró, Dániel Varró, and Katalin Friedl. Adaptive graph pattern matching for model transformations using model-sensitive search plans. In G. Karsai and G. Taentzer, editors, *GraMot 2005, International Workshop on Graph and Model Transformations*, Electronic Notes in Theoretical Computer Science, pages 191–205, 2005.
- [Zün94] Albert Zündorf. Graph pattern matching in PROGRES. In Janice E. Cuny, Hartmut Ehrig, Gregor Engels, and Grzegorz Rozenberg, editors, *TAGT*, volume 1073 of *Lecture Notes in Computer Science*, pages 454–468. Springer, 1994.

A. Proofs of Completeness

We give extensive details for the proof of Lemmas 3.6 through 3.12. In proving the main lemmas underpinning the completeness of the matching rules, we shall need a number of properties of binding bigraph structure.

A.1. Algebraic Properties of Parallel Product

The *parallel product* $G \parallel H$ of two bigraphs $G : X \rightarrow Y \uplus Z$ and $H : U \rightarrow V \uplus Z$, with $X \cap U = Y \cap V = \emptyset$, is given by taking the tensor of the place graphs and the union of the link graph maps.

The equational properties of wiring, composition and tensor product has already been investigated (Mil05, DB06); in this section we build on this foundation to state directly also a number of properties of wiring and \parallel , which shall allow us a number of convenient equational manipulations. We shall mainly use equational reasoning to prove the properties, in the process illustrating the convenience allowed by the axiomatizations and the normal form for links (cf., *loc. cit.*).

We start by giving some simple equivalent forms of the definition of the parallel product.

Lemma A.1 (Parallel product). For $Z = \{z_0, \dots, z_{n-1}\}$

$$\begin{aligned} G \parallel H &= ((\otimes_i^{|Z|} z_i / \{z_i, z'_i\}) \otimes \text{id}_Y \otimes \text{id}_V) (G \otimes ((\otimes_i^{|Z|} z'_i / z_i) \otimes \text{id}_V) H) \\ &= (\vec{z} / (\vec{z}, \vec{z}') \otimes \text{id}_Y \otimes \text{id}_V) (G \otimes (\vec{z}' / \vec{z} \otimes \text{id}_V) H) \\ &= (\sigma \otimes \text{id}_Y \otimes \text{id}_V) (G \otimes (\alpha \otimes \text{id}_V) H), \end{aligned}$$

where in the second equation we introduce some shorthand notation and in the third equation σ and α are given by the previous equations.

Proof. (Omitted) Routine from (JM04, Def. 9.13). \square

Sometimes, we shall need to split up a wiring by its inner- or outerface and analyze it in smaller parts.

When splitting up a wiring $\omega : U \uplus V \rightarrow X \uplus Y$ by its innerface or outerface, respectively, we get, for δ a suitable closure,

$$\omega = (\delta \otimes \text{id})(\sigma_0 \parallel \sigma_1) \quad \sigma_0 : U \rightarrow \quad \sigma_1 : V \rightarrow$$

respectively

$$\omega = \omega_0 \otimes \omega_1 \quad \omega_0 : \rightarrow X \quad \omega_1 : \rightarrow Y,$$

for δ , σ_0 , and σ_1 ; and ω_0 and ω_1 , which are constrained by ω , U , V , X and Y .

Those subwirings (i.e., the substitutions, closures, and wirings) of a splitting are *not* in general determined uniquely by ω , U and V , or ω , X and Y . For example, when splitting

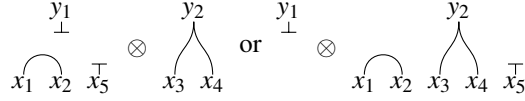
$$\omega = \begin{array}{c} y_1 \quad y_2 \\ \perp \quad \diagup \\ \text{---} \quad \text{---} \\ x_1 \quad x_2 \quad x_3 \quad x_4 \quad \overline{x_5} \end{array}$$

by $\{x_1, x_2, x_3\}$ and $\{x_4, x_5\}$, we get either

$$\begin{array}{c} y_1 \quad y_2 \\ \perp \quad \text{---} \\ \text{---} \quad \text{---} \\ x_1 \quad x_2 \quad x_3 \end{array} \parallel \parallel \begin{array}{c} y_2 \\ \text{---} \\ x_4 \quad \overline{x_5} \end{array} \quad \text{or} \quad \begin{array}{c} y_2 \\ \text{---} \\ x_1 \quad x_2 \quad x_3 \end{array} \parallel \parallel \begin{array}{c} y_2 \quad y_1 \\ \text{---} \quad \perp \\ \text{---} \\ x_4 \quad \overline{x_5} \end{array}$$

When splitting by the innerface, we are free to distribute the idle names of ω . When the wirings have no idle names (i.e., are epi) the splitting is unique. We state this formally in Lemma A.5.

When splitting by the outerface, it is closures — that introduce edges, that constitute the problem. For instance, there are four valid splittings of ω above by $\{y_1\}$ and $\{y_2\}$, corresponding to the four possible ways of distributing the two closed links between the two wirings. We might, for instance, choose to put both closures in either the lefthand or the righthand wiring:



Splittings by the outface of pure substitutions (i.e., with no closures), on the other hand, are unique; we state this formally in Lemma A.6.

We start by recording that for wirings \otimes and \parallel commute.

Lemma A.2 (Tensor and parallel product commute for wirings). For all wirings $\omega_a : \rightarrow Z \uplus Y_a$, $\omega_b : \rightarrow Z \uplus Y_b$, $\omega_c : \rightarrow X \uplus Y_c$, and $\omega_d : \rightarrow X \uplus Y_d$, with all inner faces mutually disjoint, $X \cap Z = \emptyset$, and $(Y_a \cap Y_c) = (Y_b \cap Y_d) = \emptyset$, it holds that

$$(\omega_a \parallel \omega_b) \otimes (\omega_c \parallel \omega_d) = (\omega_a \otimes \omega_c) \parallel (\omega_b \otimes \omega_d).$$

Proof. Equationally,

$$\begin{aligned} & (\omega_a \parallel \omega_b) \otimes (\omega_c \parallel \omega_d) \\ &= (\sigma_Z \otimes \text{id})(\omega_a \otimes (\alpha_Z \otimes \text{id})\omega_b) \otimes (\sigma_X \otimes \text{id})(\omega_c \otimes (\alpha_X \otimes \text{id})\omega_d) \\ &= (\sigma_Z \otimes \text{id})(\omega_a \otimes (\alpha_Z \omega_b^Z \otimes \omega_b')) \otimes (\sigma_X \otimes \text{id})(\omega_c \otimes (\alpha_X \omega_d^X \otimes \omega_d')) \\ &= (\sigma_Z \otimes \sigma_X \otimes \text{id})(\omega_a \otimes \omega_c \otimes (\alpha_Z \omega_b^Z \otimes \alpha_X \omega_d^X \otimes \omega_d' \otimes \omega_b')) \\ &= (\sigma_Z \otimes \sigma_X \otimes \text{id})(\omega_a \otimes \omega_c \otimes (\alpha_Z \otimes \alpha_X \otimes \text{id})(\omega_b^Z \otimes \omega_d^X \otimes \omega_d' \otimes \omega_b')) \\ &= (\omega_a \otimes \omega_c) \parallel (\omega_b \otimes \omega_d). \end{aligned}$$

— where the first and the last equality are instances of the third equation of Lemma A.1, and ω_b^Z , ω_b' , ω_d^X , and ω_d' arises from splitting ω_b and ω_d by the outface. \square

We state and prove Lemma A.3, a cancellation property of parallel product and idle names, only to help us prove Lemma A.4. The latter lemma tells us, that should the parallel product of two pairs of wirings with matching interfaces be equal, then those pairs were equal in the first place.

Lemma A.3. For wirings $\omega^i = X^i \rightarrow Y^i \uplus Z$ ($i \in \{0, 1\}$) with $X^0 \cap X^1 = Y^0 \cap Y^1 = \emptyset$,

$$(\omega^0 \parallel \omega^1)(X^0 \otimes \text{id}_{X^1}) = Y^0 \otimes \omega^1.$$

Proof. Equationally,

$$\begin{aligned} (\omega^0 \parallel \omega^1)(X^0 \otimes \text{id}_{X^1}) &= \omega^0 X^0 \parallel \omega^1 \\ &= (Y^0 \otimes Z) \parallel \omega^1 \\ &= Y^0 \otimes (\sigma_Z \otimes \text{id}_{Y^1})(Z \otimes (\alpha_Z \otimes \text{id}_{Y^1})\omega^1) \\ &= Y^0 \otimes (\sigma_Z \otimes \text{id}_{Y^1})(Z \otimes \alpha_Z \omega_Z^1 \otimes \omega_{Y^1}^1) \\ &= Y^0 \otimes \sigma_Z(Z \otimes \alpha_Z \omega_Z^1) \otimes \omega_{Y^1}^1 \\ &= Y^0 \otimes \alpha_Z^{-1} \alpha_Z \omega_Z^1 \otimes \omega_{Y^1}^1 \\ &= Y^0 \otimes \omega^1. \end{aligned}$$

where $\sigma_Z \stackrel{\text{def}}{=} \vec{z}/(\vec{z}, \vec{z}')$, $\alpha_Z \stackrel{\text{def}}{=} \vec{z}'/\vec{z}$ (as defined in Lemma A.1), and $\omega_Z^1 \otimes \omega_{Y^1}^2 \stackrel{\text{def}}{=} \omega^1$ is a splitting by the outface of ω^1 , s.t. $\omega_Z^1 : \rightarrow Z$ and $\omega_{Y^1}^1 : \rightarrow Y^1$.

The fifth equality resolves the composition (equationally, by iterated application of axiom L4 of (DB06)). For our particular purpose, the axiom can be instantiated to state for each $z \in Z$ that $z/(z, z')(z \otimes \text{id}_{z'}) = z/z'$. Iterating this, we have also that $\sigma_Z(Z \otimes \text{id}_{Z'}) = \alpha_Z^{-1} = \vec{z}/\vec{z}'$. \square

Lemma A.4. Given $\omega_c^i, \omega_a^i : X^i \rightarrow Y^i \uplus Z$, ($i = 0, 1$) with $X^0 \cap X^1 = Y^0 \cap Y^1 = \emptyset$, we have that

$$\omega_c^0 \parallel \omega_c^1 = \omega_a^0 \parallel \omega_a^1 \quad \text{iff} \quad \omega_c^0 = \omega_a^0 \quad \text{and} \quad \omega_c^1 = \omega_a^1.$$

Proof. (\Leftarrow) Immediate.

(\Rightarrow) Composing on both sides of the assumed equation with equal terms, we have,

$$\begin{aligned} (/Y^0 \otimes \text{id}_{Y^1 \uplus Z})(\omega_c^0 \parallel \omega_c^1)(X^0 \otimes \text{id}_{X^1}) &= (/Y^0 \otimes \text{id}_{Y^1 \uplus Z})(Y^0 \otimes \omega_c^1) = \omega_c^1, \text{ and} \\ (/Y^0 \otimes \text{id}_{Y^1 \uplus Z})(\omega_a^0 \parallel \omega_a^1)(X^0 \otimes \text{id}_{X^1}) &= (/Y^0 \otimes \text{id}_{Y^1 \uplus Z})(Y^0 \otimes \omega_a^1) = \omega_a^1. \end{aligned}$$

— and analogously for ω_c^1 and ω_a^1 (using Lemma A.3 to resolve the first equalities). \square

And now we can state and prove the two lemmas mentioned earlier, which tell us when a splitting of wiring is unique.

Lemma A.5 (Splitting by the innerface). The splitting of $\omega : U \uplus V \rightarrow$ (i.e., with $U \cap V = \emptyset$) by its innerface into $\omega = (\delta \otimes \text{id})(\sigma_0 \otimes \sigma_1)$ with $\sigma_0 : U \rightarrow$ and $\sigma_1 : V \rightarrow$ is unique (up to iso) if ω is epi.

Proof. Suppose wlog. that $\omega : U \uplus V \rightarrow Z$. We analyse the underlying link-function link_ω of ω . When ω is epi, the link-function has codomain exactly $\text{link}_\omega(U \uplus V) = Z \uplus E$, where E is the set of edges in ω . (There are no edges in E with no preimage in link_ω , as we are concerned with abstract bigraphs, which contain no such idle edges.)

The link-function of σ_0 must, for each, $u \in U$, either contain $u \mapsto \text{link}_\omega(u)$ if $\text{link}_\omega(u) \in Z$, or $u \mapsto z_u$ for a fresh name z_u if $\text{link}_\omega(u) \in E$; in the latter case, δ must also contain $z_u \mapsto \emptyset$. The same goes for σ_1 .

We can freely choose the names for transferring closed links, such as z_u ; having chosen those names, both the interfaces of σ_0 and σ_1 are also determined, since Z contains only images of $U \uplus V$. By Lemma A.4 this determines them wholly. \square

Lemma A.6 (Splitting by the outface). The splitting of $\sigma : \rightarrow X \uplus Y$ (i.e., with $X \cap Y = \emptyset$) by its outface into $\sigma = \sigma_0 \parallel \sigma_1$ with $\sigma_0 : \rightarrow X$ and $\sigma_1 : \rightarrow Y$ is unique.

Proof. Suppose wlog. that $\sigma : U \rightarrow X \uplus Y$. Since σ is a substitution its underlying link-function is defined precisely on $\text{link}_\sigma : U \rightarrow X \uplus Y$ (i.e., there are no ports or names in the link-function).

The relation link_σ^{-1} relates to each $x \in X$ and $y \in Y$ a (possibly empty) set $U_x \subseteq U$ or $U_y \subseteq U$, respectively. Since link_σ is a function, it is clear that all these sets are distinct and disjoint.

Hence, we can construct mechanically the domain U_0 of link_{σ_0} by taking the union of the preimage of the names in X , i.e., $U_0 = \text{link}_\sigma^{-1}(X)$; and this set is disjoint from $U_1 = \text{link}_\sigma^{-1}(Y)$. This procedure determines the interfaces of σ_0 and σ_1 uniquely, hence, by Lemma A.4 it determines them wholly. \square

Finally, we record a few further convenient properties of the interplay of substitutions with parallel product.

Lemma A.7. When both sides are defined, we have:

$$\sigma \parallel \sigma \alpha = \sigma(\text{id} \parallel \alpha), \quad (1)$$

$$\sigma(\omega_0 \parallel \omega_1) = \sigma \omega_0 \parallel \sigma \omega_1, \quad (2)$$

$$(\sigma_0 \parallel \sigma_1 \parallel \sigma_2)(\omega_0 \parallel \omega_1) = (\sigma_0 \parallel \sigma_2)\omega_0 \parallel (\sigma_1 \parallel \sigma_2)\omega_1. \quad (3)$$

Proof. (1) By definition of \parallel and normal form for σ (see, (DB06)).

(2) Follows easily from (1).

(3) Immediate from the earlier properties. \square

A.2. Valid Matching Sentences and Normal Forms

We start by stating two propositions, which can be derived from the normal form theorem for binding bigraphs (DB06, Theorem 1(2)).

Proposition A.8. Any discrete bigraph D of width n with local innerface can be decomposed such that

$$D = \left(\bigotimes_i^n (\widehat{\sigma}_i \otimes \text{id}) P_i \right) \pi,$$

where the P_i 's are name-discrete and prime. Any other decomposition on this form of D takes the form $(\bigotimes_i^n (\widehat{\sigma}'_i \otimes \text{id}) P'_i) \pi'$, where, for some $\widehat{\alpha}_i, \rho_i$, for all i , $P'_i = (\widehat{\alpha}_i^{-1} \otimes \text{id}) P_i \rho_i$, $(\bigotimes_i^n \rho_i) \pi' = \pi$, and $\widehat{\sigma}'_i = \widehat{\sigma}_i \widehat{\alpha}_i$.

The normal forms of name-discrete primes and free discrete molecules can be found in *loc. cit.*

One can decompose binding ions $K_{\overline{y}(\overline{x})}$ into $K_{\overline{y}(\overline{u})} \bigotimes_i^n (u_i)/(X_i)$. Such decompositions will be useful because of the following property, which is a corollary of the normal form for free discrete molecules.

In analyzing molecules, it shall be useful that the uniqueness property for free discrete molecules actually holds also for all free molecules.

Proposition A.9. For primes P and Q , if

$$(K_{\overline{y}(\overline{x})} \otimes \text{id}) P = (K_{\overline{y}(\overline{z})} \otimes \text{id}) Q,$$

then for $\alpha = \bar{x}/\bar{z}$, $K_{\bar{y}(\bar{x})}\widehat{\alpha} = K_{\bar{y}(\bar{z})}$ and $P = (\widehat{\alpha} \otimes \text{id})Q$.

Proof. (Omitted) Follows easily from normal form for primes and molecules.

□

We give a number of convenient equivalent forms for Definition 3.3 of valid matching sentences. Both are simply results of equational manipulations of the original form. In particular, the second is more compact, while the third separates global linkage from discrete bigraphs. In the proofs, we shall refer to the following Fact instead of Definition 3.3.

Fact A.10 (Valid matching sentence—with equivalent forms). Any matching sentence $\omega_a, \omega_R, \omega_C \vdash a, R \hookrightarrow C, d$, where $\omega_R : U \rightarrow Y$, for C with global outer names V , and d with global outer names Z , is *valid*, denoted $\omega_a, \omega_R, \omega_C \models a, R \hookrightarrow C, d$, iff

$$\begin{aligned} (\text{id} \otimes \omega_a)a &= (\text{id} \otimes \omega_C)(C \otimes \text{id}_Y \otimes \text{id}_Z)(\text{id}_Z \otimes (\text{id} \otimes \omega_R)R)d \\ &= (\text{id} \otimes \omega_C)((C \otimes \omega_R)R \otimes \text{id}_Z)d \\ &= (\text{id} \otimes \omega_C(\text{id}_V \otimes \omega_R \otimes \text{id}_Z))((C \otimes \text{id}_U)R \otimes \text{id}_Z)d. \end{aligned}$$

where unqualified identities are local and determined from the context.

The bigraph $((C \otimes \text{id}_U)R \otimes \text{id}_Z)d$ (i.e., the composition of the underlying discrete bigraphs of context, redex and parameter in the last form given in Fact A.10) is *not* discrete in general. Discreteness is not preserved by composition — in composing discrete D and E , local non-discrete linkage of E may be made global by composition with D . On the other hand, the expression is a ground product of primes where all edges are bound. We record some properties of these kinds of bigraphs, which follow easily from the normal form theorems based on discrete decomposition.

We start by giving a simple restatement of Proposition 3.2, eliding the details of the representation of discrete primes.

Corollary A.11. Any discrete bigraph D with local innerface of width n can be decomposed such that

$$D = \left(\bigotimes_i^n P_i \right) \pi,$$

where P_i are discrete prime. Any other decomposition on this form of D can be written as $(\bigotimes_i^n P'_i)\pi'$, where for some ρ_i , $P'_i = P_i\rho_i$ and $(\bigotimes_i^n \rho_i)\pi' = \pi$.

We call bigraphs with only bound edges *globally open*.

Definition A.12 (Globally open). G is *globally open* iff all edges in G are bound.

The following property of globally open bigraphs allows us to use the decomposition in Fact A.10 to establish a relation between the wiring of the agent and the wirings of the context, redex and parameter, even though the latter is not discretely decomposed. (We state the proposition only for ground bigraphs, as that is enough for our purposes.)

Proposition A.13 (Semi-discrete decomposition). If $(\text{id} \otimes \omega_a)d = (\text{id} \otimes \omega_b)b$, where both identities are local, d is discrete and b is globally open, then there exists a substitution σ , s.t., $\omega_a = \omega_b\sigma$ and $(\text{id} \otimes \sigma)d = b$.

Proof. Follows easily from Proposition 3.1.

By Proposition 3.1, and since b is globally open, there exists σ_e and e , where e is discrete, s.t., $b = (\text{id} \otimes \sigma_e)e$. By uniqueness of the normal form of Proposition 3.1, we have

$$\begin{aligned} \omega_a &= \omega_b\sigma_e\beta, \\ d &= (\text{id} \otimes \beta^{-1})e. \end{aligned}$$

We simply take $\sigma = \sigma_e\beta$, and we are done. □

We define products of primes with arbitrary ordering of the sites and call them *link-contained*, as they are characterized precisely by having no links that span several roots. In particular, a link-contained bigraph of width 1 is simply a prime.

Definition A.14 (Link-contained). G is *link-contained* iff G is a product of primes with arbitrary ordering of the sites.

In other words, G (of width n) can be written as $(\bigotimes_i^n P_i)\pi$, where each P_i is prime.

Link-containedness is simply a generalization of discreteness.

Lemma A.15 (Discreteness implies link-containedness).

Proof. (Omitted) Immediate from Definition A.14 and normal form for discrete. \square

A virtue of link-containedness (as opposed to discreteness) is that it is preserved by composition.

Lemma A.16 (Link-containedness is preserved by composition). If D and E are link-contained, then DE is link-contained.

Proof. (Omitted) Routine. \square

With regard to Fact A.10, Lemmas A.15 and A.16 ensures us that $((C \otimes \text{id}_U)R \otimes \text{id}_Z)d$ is link-contained.

Link-contained bigraphs share essentially all the nice properties with discrete bigraphs, that we used for establishing their normal forms. From Proposition 3.1 and Corollary A.11, we derive a normal form for link-contained bigraphs.

Corollary A.17 (Normalform for link-contained bigraphs). If G is link-contained, then G can be expressed as

$$G = \left(\bigotimes_i^n (\omega_i \otimes \text{id}) P_i \right) \pi,$$

where each P_i is prime and discrete. Further, any other expression for G on this format is of the form

$$\left(\bigotimes_i^n (\omega'_i \otimes \text{id}) Q_i \right) \pi',$$

where $(\forall i \in n)$ there exists α_i and ρ_i , s.t. $\omega_i = \omega'_i \alpha_i$, $Q_i = (\alpha_i \otimes \text{id}) P_i \rho_i$, and $(\bigotimes_i^n \rho_i) \pi' = \pi$.

Proof. (Omitted) Follows easily from the definition of link-contained and the normal forms for bigraphs and discrete bigraphs. \square

Finally, we state the following simple property of abstraction.

Lemma A.18.

$$(U)P = (U)Q \text{ iff } P = Q$$

Proof.

(\Leftarrow) Immediate.

(\Rightarrow) Nearly immediate; equationally, by composing with equal concretions:

$$P = (\ulcorner U \urcorner \otimes \text{id})(U)P = (\ulcorner U \urcorner \otimes \text{id})(U)Q = Q.$$

\square

A.3. Proofs of Lemmas 3.6 through 3.12

In this section we give the proofs for the lemmas which support the main theorem on completeness, i.e., Theorem 3.13. For ease of reading, we repeat each lemma immediately before each proof.

In the following proofs, we use equational techniques as allowed by the axiomatization of structural congruence of binding bigraphs (DB06). In particular, in manipulating terms, we shall need to introduce quite a lot of id 's on interfaces determinable from the context. We shall adopt the convention of using only unqualified local identities, when the interface is determinable, and inessential for the analysis. For global identities, we shall strive to use the metavariables introduced in Fact A.10 — i.e., we use V 's for identities on context wiring, U 's for identities on redex wiring, and Z 's for identities on parameter-wiring.

We start by proving two sublemmas, of which Lemma 3.6 will be a simple corollary. As CLOSE and PERM work solely on the three link graph and four discrete components, respectively, we proceed by proving a lemma for each of these rules.

Lemma A.19. Every valid sentence $\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \models a, R \hookrightarrow C, d$ is a consequence by PERM on a valid sentence, of equal size, of the form $\omega_{\mathbf{a}}, \omega_{\mathbf{R}}, \omega_{\mathbf{C}} \models a, S \hookrightarrow \bigotimes_i^n Q_i, e$, where each Q_i is prime (and discrete).

Proof. By Fact A.10 and Corollary A.11, C can be decomposed directly as $(\bigotimes_i^n Q_i)\pi$, while (regular) R can be decomposed as $\bigotimes_i^m P_i$ (for prime and discrete Q_i, P_i).

Applying these decompositions and the push-through lemma of (DB06) we find by standard manipulations

$$\begin{aligned} (\text{id} \otimes \omega_a)a &= (\text{id} \otimes \omega_C) \left(\left(\bigotimes_i^n Q_i \right) \pi \otimes \omega_R \right) \left(\left(\bigotimes_i^m P_i \right) \otimes \text{id}_Z \right) d, \\ &= (\text{id} \otimes \omega_C) \left(\left(\bigotimes_i^n Q_i \right) \otimes \omega_R \right) \left(\left(\bigotimes_i^m P_{\pi^{-1}(i)} \right) \otimes \text{id}_Z \right) (\bar{\pi} \otimes \text{id}_Z) d. \end{aligned}$$

Choosing $S = \bigotimes_i^m P_{\pi^{-1}(i)}$ and $e = (\bar{\pi} \otimes \text{id}_Z) d$ by Fact A.10 we have a valid sentence $\omega_a, \omega_R, \omega_C \models a, S \hookrightarrow \bigotimes_i^n Q_i, e$, which taken as the premise in the PERM yields the required sentence as conclusion. \square

Lemma A.20. Every valid sentence $\omega_a, \omega_R, \omega_C \models a, R \hookrightarrow C, d$ is a consequence by CLOSE on a valid sentence, of equal size, of the form $\sigma'_a, \sigma'_R, \sigma'_C \models a, R \hookrightarrow C, d$.

Proof. We may write ω_a, ω_R and ω_C as (by the normal form for linkage, cf. (Mil05))

$$(\text{id} \otimes /Y_a)\sigma_a \stackrel{\text{def}}{=} \omega_a, \quad (\text{id} \otimes /Y_R)\sigma_R \stackrel{\text{def}}{=} \omega_R, \quad (\text{id} \otimes /Y_C)\sigma_C \stackrel{\text{def}}{=} \omega_C.$$

(Note, that for this proof (only) we use unqualified global identities.)

We have $|Y_a| = |Y_R| + |Y_C|$; as, in particular, the number of free edges in the agent must be equal to the number of free edges in the context composed with the redex and the parameter. The discrete bigraphs d, C and R contain no free edges, so the free edges must be created entirely by ω_R and ω_C . Hence, there exists a renaming $\alpha : Y_a \rightarrow Y_R \uplus Y_C$. Assuming validity of the original sentence, we calculate

$$(\text{id} \otimes (\text{id} \otimes \alpha)\sigma_a)a = (\text{id} \otimes (\sigma_C \otimes \text{id}_{Y_R})(\text{id}_V \otimes \sigma_R \otimes \text{id}_Z))((C \otimes \text{id}_U)R \otimes \text{id}_Z)d,$$

which by Fact A.10 means that the sentence

$$(\text{id} \otimes \alpha)\sigma_a, \sigma_R, \sigma_C \otimes \text{id}_{Y_R} \models a, R \hookrightarrow C, d,$$

is valid. This sentence is on the required form, and, checking, we see that applying CLOSE to this sentence, we arrive at

$$(\text{id} \otimes /Y_C \uplus Y_R)(\text{id} \otimes \alpha)\sigma_a, (\text{id} \otimes /Y_R)\sigma_R, (\text{id} \otimes /Y_C)\sigma_C \models a, R \hookrightarrow C, d,$$

which, since by construction $/Y_C \uplus Y_R(\text{id} \otimes \alpha) = /Y_a$, is equal to

$$\omega_a, \omega_R, \omega_C \models a, R \hookrightarrow C, d.$$

\square

Lemma (3.6). Every valid sentence $\omega_a, \omega_R, \omega_C \models a, R \hookrightarrow C, d$ is provable using the CLOSE and the PERM rules on a valid sentence, of equal size, of the form $\sigma_a, \sigma_R, \sigma_C \models a, S \hookrightarrow \bigotimes_i^n P_i, e$.

Proof. Immediate by combining Lemma A.20 and Lemma A.19. \square

Lemma (3.7). Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models a, R \hookrightarrow \bigotimes_i^n P_i, d$, with each P_i prime and discrete, is provable using the PAR rule on valid sentences, of lesser or equal size, of the form $\sigma_a^0, \sigma_R^0, \sigma_C^0 \parallel \sigma_C^S \models p, S \hookrightarrow P_0, e$ and $\sigma_a^1, \sigma_R^1, \sigma_C^1 \parallel \sigma_C^S \models d', R' \hookrightarrow \bigotimes_{i=1}^n P_i, e'$. All substitutions mentioned above are required to be epi (i.e., with no idle names).

Proof. By Fact A.10 and Corollary A.11, a can be decomposed as $\bigotimes_i^n p_i$ (with discrete and prime p_i). We immediately utilize that, assuming validity of the original sentence, the width of a and $\bigotimes_i^n P_i$ must be equal.

We have, by Fact A.10 (using, as usual, unqualified local identities and introducing $C \stackrel{\text{def}}{=} \bigotimes_{i=1}^n P_i$)

$$(\text{id} \otimes \sigma_a) \bigotimes_i^n p_i = (\text{id} \otimes \sigma_C(\text{id}_{V_0} \otimes \text{id}_V \otimes \sigma_R \otimes \text{id}_Z))((P_0 \otimes C \otimes \text{id}_U)R \otimes \text{id}_Z)d.$$

By Proposition A.13, there exists a substitution σ^L , s.t.,

$$\sigma_a = \sigma_C(\text{id}_{V_0} \otimes \text{id}_V \otimes \sigma_R \otimes \text{id}_Z)\sigma^L, \quad (4)$$

$$\text{and } (\text{id} \otimes \sigma^L) \bigotimes_i^n p_i = ((P_0 \otimes C \otimes \text{id}_U)R \otimes \text{id}_Z)d. \quad (5)$$

We can partition the redex according to the innerfaces of P_0 and C ; and then partition the parameter according to the partitioning of the redex. From (5) we can derive

$$\begin{aligned} (\text{id} \otimes \sigma^{\mathbf{L}}) \bigotimes_{i=1}^n p_i &= ((P_0 \otimes \text{id}_{U_0})R_0 \otimes \text{id}_{Z_0})d_0 \otimes \\ &\quad ((C \otimes \text{id}_{U'}) \bigotimes_{i=1}^m R_i \otimes \text{id}_{Z'}) \bigotimes_{i=1}^l d_i, \end{aligned} \quad (6)$$

introducing a few more convenient metavariables: For the partitioned parts of the redex and parameter, we let $\bigotimes_{i=1}^m R_i \stackrel{\text{def}}{=} R$ and $\bigotimes_{i=1}^l d_i \stackrel{\text{def}}{=} d$ (where each R_i and d_i is a product of primes). Also, in the following, we shall use the shorthands $R' \stackrel{\text{def}}{=} \bigotimes_{i=1}^m R_i$ and $d' \stackrel{\text{def}}{=} \bigotimes_{i=1}^l d_i$.

From (6) we see that we must be able to partition $\sigma^{\mathbf{L}}$ equally. The substitution must be on the form $\sigma^{\mathbf{L}} = (\sigma_{\mathbf{P}_0}^{\mathbf{L}} \otimes \sigma_{\mathbf{R}_0}^{\mathbf{L}} \otimes \sigma_{\mathbf{d}_0}^{\mathbf{L}}) \otimes (\sigma_{\mathbf{C}}^{\mathbf{L}} \otimes \sigma_{\mathbf{R}'}^{\mathbf{L}} \otimes \sigma_{\mathbf{d}'}^{\mathbf{L}})$, as we can split $\sigma^{\mathbf{L}}$ according to the outerfaces of $((P_0 \otimes \text{id}_{U_0})R_0 \otimes \text{id}_{Z_0})d_0$ and $((C \otimes \text{id}_{U'})R' \otimes \text{id}_{Z'})d'$. (This splitting is unique; as are all the other splittings of wiring in this lemma, since we work only on epi substitutions — cf. Lemma A.5 and Lemma A.6.)

By Corollary A.17, as the bigraphs are ground (and hence, there are no permutations to consider), we find

$$(\text{id} \otimes \sigma_{\mathbf{P}_0}^{\mathbf{L}} \otimes \sigma_{\mathbf{R}_0}^{\mathbf{L}} \otimes \sigma_{\mathbf{d}_0}^{\mathbf{L}}) p_0 = ((P_0 \otimes \text{id}_{U_0})R_0 \otimes \text{id}_{Z_0})d_0 \quad (7)$$

$$\text{and } (\text{id} \otimes \sigma_{\mathbf{C}}^{\mathbf{L}} \otimes \sigma_{\mathbf{R}'}^{\mathbf{L}} \otimes \sigma_{\mathbf{d}'}^{\mathbf{L}}) \bigotimes_{i=1}^n p_i = ((C \otimes \text{id}_{U'})R' \otimes \text{id}_{Z'})d'. \quad (8)$$

We can also split $\sigma_{\mathbf{a}}$ by its innerface, which it shares with $\sigma^{\mathbf{L}}$ (cf. (4)), and define

$$\left(\sigma_{\mathbf{a}}^{\mathbf{P}_0} \parallel \sigma_{\mathbf{a}}^{\mathbf{R}_0} \parallel \sigma_{\mathbf{a}}^{\mathbf{d}_0} \right) \parallel \left(\sigma_{\mathbf{a}}^{\mathbf{C}} \parallel \sigma_{\mathbf{a}}^{\mathbf{R}'} \parallel \sigma_{\mathbf{a}}^{\mathbf{d}'} \right) \stackrel{\text{def}}{=} \sigma_{\mathbf{a}}.$$

(In the following we also use $\sigma_{\mathbf{a}}^{\mathbf{R}} \stackrel{\text{def}}{=} \sigma_{\mathbf{a}}^{\mathbf{R}_0} \parallel \sigma_{\mathbf{a}}^{\mathbf{R}'}$.)

Equally, we can split $\sigma_{\mathbf{R}}$ by its innerface into $\sigma_{\mathbf{R}_0} \parallel \sigma_{\mathbf{R}'}$ (corresponding to the outerface of $\sigma_{\mathbf{R}_0}^{\mathbf{L}}$ and $\sigma_{\mathbf{R}'}^{\mathbf{L}}$, respectively). Finally, we split $\sigma_{\mathbf{C}}$ by its innerface (cf. (4)), to define

$$\sigma_{\mathbf{C}}^{\mathbf{P}_0} \parallel \sigma_{\mathbf{C}}^{\mathbf{d}_0} \parallel \sigma_{\mathbf{C}}^{\mathbf{R}} \parallel \sigma_{\mathbf{C}}^{\mathbf{C}} \parallel \sigma_{\mathbf{C}}^{\mathbf{d}'} \stackrel{\text{def}}{=} \sigma_{\mathbf{C}}.$$

We cannot immediately split $\sigma_{\mathbf{C}}^{\mathbf{R}}$ into two, as we could for $\sigma_{\mathbf{a}}^{\mathbf{R}}$. This part of the context-wiring needs special care. Now (4) can be expressed like this (rearranging terms slightly to compose matching wirings)

$$\begin{aligned} \sigma_{\mathbf{a}}^{\mathbf{P}_0} \parallel \sigma_{\mathbf{a}}^{\mathbf{d}_0} \parallel \sigma_{\mathbf{a}}^{\mathbf{R}} \parallel \sigma_{\mathbf{a}}^{\mathbf{C}} \parallel \sigma_{\mathbf{a}}^{\mathbf{d}'} &= (\sigma_{\mathbf{C}}^{\mathbf{P}_0} \parallel \sigma_{\mathbf{C}}^{\mathbf{d}_0} \parallel \sigma_{\mathbf{C}}^{\mathbf{R}} \parallel \sigma_{\mathbf{C}}^{\mathbf{C}} \parallel \sigma_{\mathbf{C}}^{\mathbf{d}'}) \circ \\ &\quad \sigma_{\mathbf{P}_0}^{\mathbf{L}} \otimes \sigma_{\mathbf{d}_0}^{\mathbf{L}} \otimes (\sigma_{\mathbf{R}_0}^{\mathbf{L}} \parallel \sigma_{\mathbf{R}'}^{\mathbf{L}}) \otimes \sigma_{\mathbf{C}}^{\mathbf{L}} \otimes \sigma_{\mathbf{d}'}^{\mathbf{L}} \\ &= \sigma_{\mathbf{C}}^{\mathbf{P}_0} \sigma_{\mathbf{P}_0}^{\mathbf{L}} \parallel \sigma_{\mathbf{C}}^{\mathbf{d}_0} \sigma_{\mathbf{d}_0}^{\mathbf{L}} \parallel \sigma_{\mathbf{C}}^{\mathbf{R}} (\sigma_{\mathbf{R}_0}^{\mathbf{L}} \parallel \sigma_{\mathbf{R}'}^{\mathbf{L}}) \parallel \sigma_{\mathbf{C}}^{\mathbf{C}} \sigma_{\mathbf{C}}^{\mathbf{L}} \parallel \sigma_{\mathbf{C}}^{\mathbf{d}'} \sigma_{\mathbf{d}'}^{\mathbf{L}}. \end{aligned}$$

We note that by Lemma A.4, as each pair of wirings of the parallel product have equal interfaces, they are equal (i.e., $\sigma_{\mathbf{a}}^{\mathbf{P}_0} = \sigma_{\mathbf{C}}^{\mathbf{P}_0} \sigma_{\mathbf{P}_0}^{\mathbf{L}}$, $\sigma_{\mathbf{a}}^{\mathbf{d}_0} = \sigma_{\mathbf{C}}^{\mathbf{d}_0} \sigma_{\mathbf{d}_0}^{\mathbf{L}}$, ...).

We now consider the substitution working on the redex stemming from the context, $\sigma_{\mathbf{C}}^{\mathbf{R}}$, which needs a little extra care. We have, by the arguments above, in particular $\sigma_{\mathbf{a}}^{\mathbf{R}} = \sigma_{\mathbf{C}}^{\mathbf{R}} (\sigma_{\mathbf{R}_0}^{\mathbf{L}} \parallel \sigma_{\mathbf{R}'}^{\mathbf{L}})$.

Splitting $\sigma_{\mathbf{C}}^{\mathbf{R}}$ by the outerfaces of $\sigma_{\mathbf{R}_0}$ and $\sigma_{\mathbf{R}'}$, we get

$$\sigma_{\mathbf{a}}^{\mathbf{R}} = \sigma_{\mathbf{a}}^{\mathbf{R}_0} \parallel \sigma_{\mathbf{a}}^{\mathbf{R}'} = (\sigma_{\mathbf{C}}^{\mathbf{R}_0} \parallel \sigma_{\mathbf{C}}^{\mathbf{R}'}) (\sigma_{\mathbf{R}_0}^{\mathbf{L}} \parallel \sigma_{\mathbf{R}'}^{\mathbf{L}}),$$

as, in general, $\sigma_{\mathbf{R}_0}$ and $\sigma_{\mathbf{R}'}$ can share names in their outerface; $\sigma_{\mathbf{C}}^{\mathbf{R}'}$ works on those shared names.

For ease of notation, we break our metavariable conventions for substitutions temporarily, and introduce $\phi_{\mathbf{C}}^{\mathbf{R}} = \sigma_{\mathbf{C}}^{\mathbf{R}_0} \parallel \sigma_{\mathbf{C}}^{\mathbf{R}'}$ and $\psi_{\mathbf{C}}^{\mathbf{R}} = \sigma_{\mathbf{C}}^{\mathbf{R}_0} \parallel \sigma_{\mathbf{C}}^{\mathbf{R}'}$. Applying Lemma A.7(2), we have

$$\sigma_{\mathbf{a}}^{\mathbf{R}_0} \parallel \sigma_{\mathbf{a}}^{\mathbf{R}'} = (\phi_{\mathbf{C}}^{\mathbf{R}} \sigma_{\mathbf{R}_0}^{\mathbf{L}} \parallel \psi_{\mathbf{C}}^{\mathbf{R}} \sigma_{\mathbf{R}'}^{\mathbf{L}}),$$

and, applying Lemma A.4, we find that

$$\sigma_a^{R_0} = \phi_C^R \sigma_{R_0}^L \sigma_{R_0}^L, \quad (9)$$

$$\sigma_a^{R'} = \psi_C^R \sigma_{R'}^L \sigma_{R'}^L. \quad (10)$$

And now, finally, we are set to utilize what we have learnt from these somewhat tedious symbol manipulations. Composing on both sides of (7) with $\text{id} \otimes (\sigma_C^{P_0} \parallel \phi_C^R \sigma_{R_0} \parallel \sigma_C^{d_0})$, and using the equalities for the substitutions that we have found above, we derive

$$\begin{aligned} & \left(\text{id} \otimes (\sigma_C^{P_0} \parallel \phi_C^R \sigma_{R_0} \parallel \sigma_C^{d_0}) \right) (\text{id} \otimes \sigma_{P_0}^L \otimes \sigma_{R_0}^L \otimes \sigma_{d_0}^L) p_0 = \\ & \left(\text{id} \otimes (\sigma_C^{P_0} \parallel \phi_C^R \sigma_{R_0} \parallel \sigma_C^{d_0}) \right) ((P_0 \otimes \text{id}_{U_0}) R_0 \otimes \text{id}_{Z_0}) d_0 \\ \iff & \left(\text{id} \otimes (\sigma_a^{P_0} \parallel \sigma_a^{R_0} \parallel \sigma_a^{d_0}) \right) p_0 = \left(\text{id} \otimes ((\sigma_C^{P_0} \parallel \phi_C^R \parallel \sigma_C^{d_0}) (\text{id}_{V_0} \otimes \sigma_{R_0} \otimes \text{id}_{Z_0})) \right) \circ \\ & ((P_0 \otimes \text{id}_{U_0}) R_0 \otimes \text{id}_{Z_0}) d_0 \end{aligned}$$

By analogous manipulations from (8) and, using in particular (10), we can deduce that we have valid sentences

$$\sigma_C^{P_0} \parallel \sigma_a^{R_0} \parallel \sigma_a^{d_0}, \sigma_{R_0}, \sigma_C^{P_0} \parallel \sigma_C^{d_0} \parallel \sigma_C^{R_0} \models p_0, R_0 \hookrightarrow P_0, d_0,$$

and

$$\sigma_a^C \parallel \sigma_a^{R'} \parallel \sigma_a^{d'}, \sigma_{R'}, \sigma_C^C \parallel \sigma_C^{d'} \parallel \sigma_C^{R'} \models \bigotimes_{i=1}^n p_i, R' \hookrightarrow \bigotimes_{i=1}^n P_i, d',$$

which by PAR yields the original sentence. \square

Lemma (3.8). Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models a, R \hookrightarrow \text{id}_\varepsilon, d$ is provable using PAR and WIRING-AXIOM.

Proof. Since $C = \text{id}_\varepsilon$, a must have width 0, hence also $a = \text{id}_\varepsilon$ the only discrete bigraph with local innerface (in this case, actually ground innerface) of width 0. Similarly, R and d must have width 0, hence be id_ε . We analyze the wirings, in turn:

Agent By Fact A.10, the agent is expressible as $(\sigma_a \otimes \text{id}_\varepsilon) \text{id}_\varepsilon$, hence $\sigma_a = Y$ for some Y .

Context Equally, by Fact A.10, the context is $(\sigma_C \otimes \text{id}_\varepsilon) (\text{id}_\varepsilon \otimes \text{id}_U)$ (for some names U stemming from the redex (wiring)). Hence, $\sigma_C : U \rightarrow Y$.

Redex And finally, also by Fact A.10, redex is $(\sigma_R \otimes \text{id}_\varepsilon) \text{id}_\varepsilon$, hence $\sigma_R : \varepsilon \rightarrow U = U$.

By the arguments above, the original sentence must be on the form,

$$Y, U, \sigma_C \models \text{id}_\varepsilon, \text{id}_\varepsilon \hookrightarrow \text{id}_\varepsilon, \text{id}_\varepsilon \quad (\sigma_C : U \rightarrow Y).$$

By induction on the size of Y it is immediate that this sentence is derivable by $|Y| - 1$ applications of PAR from sentences, which are instances of WIRING-AXIOM. \square

Note A.21. Iterating Lemma 3.7 allows us to break any product of discrete primes in context and agent into prime parts, resulting in sentences of the form

$$\sigma_a, \sigma_R, \sigma_C \models p, R \hookrightarrow P, d,$$

for (discrete) primes p, P and epi substitutions $\sigma_a, \sigma_R, \sigma_C$. We treat in Lemma 3.7 only wirings with no idle names; we handle the idle names by (iterated) application of Lemma 3.8, where there are no primes in the context or agent.

Hence, in the following lemmas all wiring is epi substitutions, and we are guaranteed that any splitting of wiring is unique.

Lemma (3.9). Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models p, R \hookrightarrow P, d$, with p and P prime and discrete, is provable using the LSUB rule on a valid sentence, of lesser or equal size, of the form $\sigma_a', \sigma_R', \sigma_C' \models p', R' \hookrightarrow P', d$, where p' and P' are discrete free primes. All substitutions mentioned above are required to be epi (i.e., with no idle names).

Working on (12) we get (applying the push-through-lemma (DB06)),

$$\begin{aligned}
& (\text{id} \otimes \sigma^{\mathbf{L}}) (\text{merge} \otimes \text{id}) \bigotimes_i^k m_i \\
&= (\text{merge} \otimes \text{id}) \left(\left(\left(\bigotimes_i^n \alpha_i^\top \right) \otimes \left(\bigotimes_i^m M_i \right) \otimes \text{id}_U \right) \left(\bigotimes_i^l P_{\pi^{-1}(i)} \right) \otimes \text{id}_Z \right) \bar{\pi} d \\
&= (\text{merge} \otimes \text{id}) \left(\left(\left(\bigotimes_i^n (\Gamma \alpha_i^\top \otimes \text{id}) P_{\pi^{-1}(i)} \right) \otimes \left(\bigotimes_i^m (M_i \otimes \text{id}) S_i \right) \otimes \text{id}_Z \right) \bar{\pi} d \right. \\
&= (\text{merge} \otimes \text{id}) \left(\left(\left(\bigotimes_i^n ((\Gamma \alpha_i^\top \otimes \text{id}) P_{\pi^{-1}(i)} \otimes \text{id}) e_i \right) \otimes \bigotimes_i^m ((M_i \otimes \text{id}) S_i \otimes \text{id}) f_i \right), \right. \\
&= (\text{merge} \otimes \text{id}) \left(\left(\bigotimes_i^n t_i \right) \otimes \bigotimes_i^m u_i \right), \tag{13}
\end{aligned}$$

where we define

$$\bigotimes_i^m S_i \stackrel{\text{def}}{=} \bigotimes_{i=n}^l P_{\pi^{-1}(i)}, \quad \left(\bigotimes_i^n e_i \right) \otimes \bigotimes_i^m f_i \stackrel{\text{def}}{=} \bar{\pi} d,$$

$$\forall i \in n \quad t_i \stackrel{\text{def}}{=} \left((\Gamma \alpha_i^\top \otimes \text{id}) P_{\pi^{-1}(i)} \otimes \text{id} \right) e_i, \text{ and}$$

$$\forall i \in m \quad u_i \stackrel{\text{def}}{=} ((M_i \otimes \text{id}) S_i \otimes \text{id}) f_i.$$

Each e_i , S_i and f_i are determined by the innerfaces of the corresponding $P_{\pi^{-1}(i)}$, M_i and S_i , respectively.

Since $\sigma^{\mathbf{L}}$ shares global outface with the product of the t_i 's and u_i 's, we can partition it according to the outfaces of those primes.

$$\left(\bigotimes_i^n \sigma_i^{\mathbf{t}} \right) \otimes \bigotimes_i^m \sigma_i^{\mathbf{u}} = \sigma \tag{14}$$

We do not want to break the redex and parameter entirely into molecules in this step. Instead, as each m_i represent a top-level node of the agent, it is easily seen that we can permute and partition the m_i 's into n products, p_i , and m molecules, n_i , which match the place-graph structure in each t_i and u_i , respectively:

$$\left(\bigotimes_i^n p_i \right) \otimes \bigotimes_i^m n_i \stackrel{\text{def}}{=} (\rho \otimes \text{id}) \bigotimes_i^k m_i \tag{15}$$

for some permutation ρ . Since $\sigma^{\mathbf{L}}$ was also partitioned according to the outfaces of t_i and u_i , the outface of each p_i will match the innerface of each $\sigma_i^{\mathbf{t}}$, and the outface of each n_i will match the innerface of each $\sigma_i^{\mathbf{u}}$.

More formally, we can easily prove a little lemma for ground molecules and primes, saying that

$$(\text{merge} \otimes \text{id}) \bigotimes_i^n m_i = (\text{merge} \otimes \text{id}) \bigotimes_i^m p_i$$

iff (for some ρ, k_i)

$$\forall i \in m \quad p_i = (\text{merge} \otimes \text{id}) \left(\bigotimes_j^{k_i} m_{\rho(j)} \right).$$

To underline the point: When we have split one of the products under a *merge* wholly into molecules, we can express each prime p_i by each of the molecules m_i that match it.

Consequentially, by combining (14) and (15), we find that

$$(\forall i \in n) (\sigma_i^t \otimes \text{id}) p_i = t_i \quad (16)$$

$$\text{and } (\forall i \in m) (\sigma_i^u \otimes \text{id}) n_i = u_i \quad (17)$$

Now we concern ourselves with the wirings of equation (11). We split σ_a by the innerface according to the inner-faces of the σ_i^t 's and σ_i^u 's, to get

$$\sigma_a = \left(\prod_i^n \sigma_{i,a}^t \right) \parallel \left(\prod_i^m \sigma_{i,a}^u \right).$$

We also split $\sigma_C(\sigma_R \otimes \text{id}_V \otimes \text{id}_Z)$ accordingly

$$\begin{aligned} & \sigma_C(\sigma_R \otimes \text{id}_V \otimes \text{id}_Z) \left(\left(\prod_i^n \sigma_i^t \right) \otimes \left(\prod_i^m \sigma_i^u \right) \right) \\ &= \sigma_C \left(\left(\prod_i^n \sigma_{i,R}^t \right) \parallel \left(\prod_i^m \sigma_{i,R}^u \right) \otimes \text{id}_V \otimes \text{id}_Z \right) \left(\left(\prod_i^n \sigma_i^t \right) \otimes \left(\prod_i^m \sigma_i^u \right) \right) \\ &= \left(\left(\prod_i^n \sigma_{i,C}^t(\sigma_{i,R}^t \otimes \text{id}) \right) \parallel \left(\prod_i^m \sigma_{i,C}^u(\sigma_{i,R}^u \otimes \text{id}) \right) \right) \left(\left(\prod_i^n \sigma_i^t \right) \otimes \left(\prod_i^m \sigma_i^u \right) \right) \end{aligned} \quad (18)$$

in the first step splitting σ_R , s.t.

$$\sigma_R = \left(\prod_i^n \sigma_{i,R}^t \right) \parallel \left(\prod_i^m \sigma_{i,R}^u \right),$$

and in the second step using Lemma A.7(3) (iterated) to split shared wiring in σ_C into $n + m$ substitutions according to $\sigma_{i,R}^t$ and $\sigma_{i,R}^u$.

As each $\sigma_{i,a}^t$ and $\sigma_{i,a}^u$ has the same interfaces as $\sigma_{i,C}^t(\sigma_{i,R}^t \otimes \text{id})$ and $\sigma_{i,C}^u(\sigma_{i,R}^u \otimes \text{id})$, respectively, by Lemma A.4 they are equal. By equational manipulations identical to those concluding the proof of Lemma 3.7, we can infer from (16), (17), and (18), that,

$$(\forall i \in n) (\sigma_{i,a}^t \otimes \text{id}) p_i = (\sigma_{i,C}^t(\sigma_{i,R}^t \otimes \text{id}) \otimes \text{id}) t_i$$

$$\text{and } (\forall i \in m) (\sigma_{i,a}^u \otimes \text{id}) n_i = (\sigma_{i,C}^u(\sigma_{i,R}^u \otimes \text{id}) \otimes \text{id}) u_i.$$

We check and find, that we have $n + m$ valid sentences which by PAR (iterated) yields the original sentence by a single application of MERGE.

Finally, we note that each sentence corresponding to the first n equations,

$$\sigma_{i,a}^t, \sigma_{i,R}^t, \sigma_{i,C}^t \models p_i, P_{\pi^{-1}(i)} \xrightarrow{\lceil \alpha_i \rceil} e_i,$$

is a consequence by SWITCH of the sentence,

$$\sigma_{i,a}^t, \text{id}_\varepsilon, \sigma_{i,C}^t \left(\alpha_i \sigma_{\pi^{-1}(i)} \otimes \sigma_{i,R}^t \otimes \text{id} \right) \models p_i, \text{id} \xleftrightarrow{} P_{\pi^{-1}(i)}^N, e_i$$

for $P_i = (\widehat{\sigma}_i \otimes \text{id}) (Y_i) P_i^N$; where $P_{\pi^{-1}(i)}^N$ is discrete, free and prime. \square

Lemma (3.11). Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models m, R \xleftrightarrow{} M, d$, with m and M free discrete molecules, is provable using the ION rule on a valid sentence $\sigma'_a, \sigma'_R, \sigma'_C \models p, R \xleftrightarrow{} P, d$, of lesser size, where p and P are discrete primes. All substitutions mentioned above are required to be epi (i.e., with no idle names).

Proof. By Fact A.10, and the normal form for free discrete molecules (cf. (DB06)), we can express m as

$$m = \left(K_{y(\bar{x})} \otimes \text{id} \right) q,$$

and M as

$$M = \left(K_{\bar{u}(\bar{w})} \otimes \text{id} \right) Q,$$

where q and Q are namediscrete and prime.⁵

Assuming validity of the original sentence, by Fact A.10 and Proposition A.13, there exists a substitution σ^L , s.t.,

$$\sigma_a = \sigma_C(\text{id}_V \otimes \sigma_R \otimes \text{id}_Z) \sigma^L. \quad (19)$$

$$\text{and } (\text{id} \otimes \sigma^L) \left(K_{\vec{y}(\vec{X})} \otimes \text{id} \right) q = \left(\left(\left(K_{\vec{u}(\vec{W})} \otimes \text{id} \right) Q \otimes \text{id}_U \right) R \otimes \text{id}_Z \right) d. \quad (20)$$

Since the K -node on the righthand-side is wired discretely to \vec{u} 's, the lefthand-side must match this. Hence, we must have

$$\sigma^L = \vec{u}/\vec{y} \otimes \sigma_1^L.$$

We define,

$$\widehat{\sigma}_{(\vec{X})} \stackrel{\text{def}}{=} \bigotimes_i^{|\vec{X}|} (x_i)/(X_i) \quad \text{and} \quad \widehat{\phi}_{(\vec{W})} \stackrel{\text{def}}{=} \bigotimes_i^{|\vec{W}|} (x_i)/(W_i),$$

such that we can decompose the ions, and express (20) as,

$$\left(K_{\vec{u}(\vec{x})} \otimes \text{id} \right) \left(\widehat{\sigma}_{(\vec{X})} \otimes \sigma_1^L \right) q = \left(K_{\vec{u}(\vec{x})} \otimes \text{id} \right) \left(\left(\left(\widehat{\phi}_{(\vec{W})} \otimes \text{id} \right) Q \otimes \text{id}_U \right) R \otimes \text{id}_Z \right) d. \quad (21)$$

Applying Proposition A.9, we have,

$$\left(\widehat{\sigma}_{(\vec{X})} \otimes \sigma_1^L \right) q = \left(\left(\left(\widehat{\phi}_{(\vec{W})} \otimes \text{id} \right) Q \otimes \text{id}_U \right) R \otimes \text{id}_Z \right) d. \quad (22)$$

We split σ_a and σ_C into the wiring linked to the K -node (i.e., the \vec{y} 's, and the \vec{u} 's, respectively) and a remainder.

$$\sigma_a^K \parallel \sigma_a^r \stackrel{\text{def}}{=} \sigma_a,$$

$$\text{and } \sigma_C^K \vec{u}/\vec{y} \parallel \sigma_C^r (\sigma_R \otimes \text{id}) \sigma_1^L \stackrel{\text{def}}{=} \sigma_C (\text{id}_V \otimes \sigma_R \otimes \text{id}_Z) (\vec{u}/\vec{y} \otimes \sigma_1^L).$$

From (19), with the help of Lemma A.4, then also, as the interfaces of the corresponding substitutions match,

$$\sigma_a^K = \sigma_C^K \vec{u}/\vec{y},$$

$$\text{and } \sigma_a^r = \sigma_C^r (\text{id}_V \otimes \sigma_R \otimes \text{id}_Z) \sigma_1^L.$$

Composing on both sides of (22) with $(\text{id} \otimes \sigma_C^r (\text{id}_V \otimes \sigma_R \otimes \text{id}_Z))$ and using the equalities defined above, we find,

$$\begin{aligned} (\text{id} \otimes \sigma_a^r) \left(\widehat{\sigma}_{(\vec{X})} \otimes \text{id} \right) q \\ = (\text{id} \otimes \sigma_C^r (\text{id}_V \otimes \sigma_R \otimes \text{id}_Z)) \left(\left(\left(\widehat{\phi}_{(\vec{W})} \otimes \text{id} \right) Q \otimes \text{id}_U \right) R \otimes \text{id}_Z \right) d. \end{aligned}$$

Choosing $\sigma'_a = \sigma_a^r$, $\sigma'_R = \sigma_R$, $\sigma'_C = \sigma_C^r$, $p = (\widehat{\sigma}_{(\vec{X})} \otimes \text{id}) q$, and

$$P = \left(\left(\left(\widehat{\phi}_{(\vec{W})} \otimes \text{id} \right) Q \otimes \text{id}_U \right) R \otimes \text{id}_Z \right) d,$$

we have a valid sentence, which by ION yields the original sentence. \square

Lemma (3.12). Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models p, \text{id} \hookrightarrow P, e$, with p and P free discrete primes, is provable using the MERGE and PAR (iterated) rules on valid sentences of equal or lesser size, which are either instances of rule PRIME-AXIOM or of the form $\sigma'_a, \sigma'_R, \sigma'_M \models m, R \hookrightarrow M, d$. All substitutions mentioned above are required to be epi (i.e., with no idle names).

Proof. (Sketch) The proof is similar to the proof of Lemma 3.10, but simpler as the redex is a (local) identity. This also implies $\sigma_R = \text{id}_e$, since for some local identity $(\text{id} \otimes \sigma_R)R$ must be defined. For the concretions in P , instead of arriving at sentences which are derivable by SWITCH, each such sentence is an instance of PRIME-AXIOM. We treat these basecases in more detail below.

⁵ Note, that for this proof only, we break our convention of only eliding interfaces on local identities. The identities introduced in the expressions for m and M are global, but as they are inessential for the analysis, to ease the notational clutter, we shall elide the interfaces for these two identities.

The analysis for the wirings is simplified as $\sigma_{\mathbf{R}}$ is an identity. We find (focusing only on the wiring concerned with the concretions of P),

$$(\forall i \in n) \sigma_{i,\mathbf{a}}^{\mathbf{t}} = \sigma_{i,\mathbf{C}}^{\mathbf{t}} \sigma_i^{\mathbf{t}}. \quad (23)$$

Performing an analysis analogous to that in the proof of Lemma 3.10 we have (instead of (16))

$$(\forall i \in n) (\sigma_i^{\mathbf{t}} \otimes \text{id}) p_i = (\alpha_i \otimes \text{id} \otimes \text{id}_{Z_i}) (\ulcorner U_i \urcorner \otimes \text{id}_{Z_i}) e_i, \quad (24)$$

for renamings $\alpha_i : X_i \rightarrow U_i$, and primes $e_i : \langle Z_i \uplus U_i \rangle$ stemming from the parameter.

From (24), we can derive an expression for e_i in terms of p_i . For each $i \in n$, we have

$$\begin{aligned} & (\sigma_i^{\mathbf{t}} \otimes \text{id}) p_i = (\alpha_i \otimes \text{id} \otimes \text{id}_{Z_i}) (\ulcorner U_i \urcorner \otimes \text{id}_{Z_i}) e_i \\ \iff & (\alpha_i^{-1} \otimes \text{id} \otimes \text{id}_{Z_i}) (\sigma_i^{\mathbf{t}} \otimes \text{id}) p_i = (\ulcorner U_i \urcorner \otimes \text{id}_{Z_i}) e_i \\ \iff & (X_i) (\alpha_i^{-1} \otimes \text{id} \otimes \text{id}_{Z_i}) (\sigma_i^{\mathbf{t}} \otimes \text{id}) p_i = e_i \end{aligned} \quad (25)$$

Since we know e_i is discrete, from (25) we see that $\sigma_i^{\mathbf{t}}$ must be a renaming on all names not linked to U_i (to be localized by the abstraction). Hence, for each $i \in n$,

$$\sigma_i^{\mathbf{t}} = \sigma_i^{\mathbf{t}'} \otimes \beta_i,$$

for $\beta_i : \rightarrow Z_i$ and $\sigma_i^{\mathbf{t}'} : \rightarrow U_i$.

By composing on both sides of (24) with $(\sigma_{i,\mathbf{C}}^{\mathbf{t}} \otimes \text{id})$ and utilizing the forms for e_i from (25)

$$\begin{aligned} (\forall i \in n) (\sigma_{i,\mathbf{C}}^{\mathbf{t}} (\sigma_i^{\mathbf{t}'} \otimes \beta_i) \otimes \text{id}) p_i \\ = (\sigma_{i,\mathbf{C}}^{\mathbf{t}} \otimes \text{id}) (\ulcorner \alpha_i \urcorner \otimes \text{id}_{Z_i}) (X_i) (\alpha_i^{-1} \otimes \text{id} \otimes \text{id}_{Z_i}) (\sigma_i^{\mathbf{t}} \otimes \text{id}) p_i \end{aligned} \quad (26)$$

To be strict, we also need to refer to equation (23) to see that $\sigma_{i,\mathbf{C}}^{\mathbf{t}} (\sigma_i^{\mathbf{t}'} \otimes \beta_i)$ is, in fact, wiring stemming from the agent. In other words, (23) gives an equation of the agent wiring in terms of the wiring from the context and parameter.

The n equations stated in (26), means that we have n valid sentences on the form

$$\sigma_{i,\mathbf{C}}^{\mathbf{t}} (\sigma_i^{\mathbf{t}'} \otimes \beta_i), \text{id}_{\varepsilon}, \sigma_{i,\mathbf{C}}^{\mathbf{t}} \vDash p_i, \text{id}_{(X_i)} \hookrightarrow \ulcorner \alpha_i \urcorner, (X_i) (\alpha_i^{-1} \otimes \text{id} \otimes \text{id}_{Z_i}) (\sigma_i^{\mathbf{t}} \otimes \text{id}) p_i.$$

It is easily verified that those sentences are instances of PRIME-AXIOM (choosing, in particular, τ equal to $\alpha_i^{-1} \sigma_i^{\mathbf{t}'}$).

□