BiLog: Spatial Logics for Bigraphs

G□□□□□□□ C□□□□□□□, D□□□□□□ M□□□□□□□□ and V□□□□□□□□ S□□□□□□

A□□□□□□□ amiano

- In ‘separation’ logics [23], it is used to reason about dynamic update of heap-like structures, and it is *strong* in that it forces names of resources in separated components to be disjoint. As a consequence, term composition is usually partially defined.

- In static spatial logics (e.g. for trees [3], graphs [5] or trees with hidden names [6]), the separation/composition does not require any constraint on terms, and names are usually shared between separated parts.

- Also in dynamic spatial logics (e.g. for ambients [7] or π-calculus [1]) the separation is intended only for locations in space.

Context tree logic, introduced in [4], integrates the first approach above with a spatial logic for trees. The result is a logic able to express properties of tree-shaped structures (and contexts) with pointers, and it is used as an assertion language for Hoare-style program specifications in a tree memory model. Essentially Spatial Logic uses the structure of the model to give semantics.

Bigraphs [16, 18] are an emerging model for structures in global computing, that can be instantiated to model several well-known examples, including λ-calculus [21], CCS [22], π-calculus [16], ambients [17] and Petri nets [20]. Bigraphs consist essentially of two graphs sharing the same nodes. The first graph, the *place gr Tf(xts))]TJ/F8tion gr Tf(xts)8tion)-54ace any constraina336(in)*

This describes two PC with different names, *a* and *b*, sharing a link on a distinct name *c*, Rdainct7el,6(moden)1ls(,)-295e.g. 7.l6(a 7.l6(communicpat(on)-l6(channel. 7417(Nr
)]TJ/d)-270(nan)1[(,kTJ/35.156atJ/35.1c 16635.13 Tf35.1()ll 0 T35.13yJ/35.1composi95e.TJ/628 9.uTJ/3

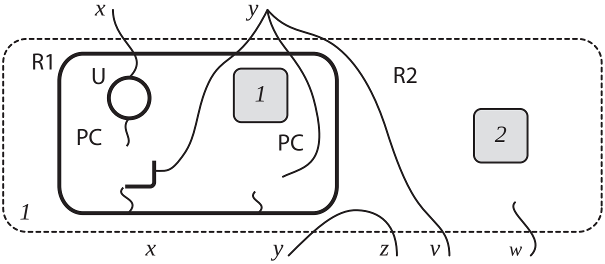

FIGURE 1. A bigraph $G : \langle 2, \{x, y, z, v, w\}\rangle \rightarrow \langle 1, \{x, y\}\rangle$.

graph. Place graphs express locality, that is the physical arrangement of the nodes. Link graphs are hyper-graphs and formalise connections among nodes. The orthogonality of the two structures dictates that nestings impose no constrain upon interconnections.

The bigraph G of Fig. 1 represents a system where people and things interact. We imagine two offices with employees logged on PCs. Every entity is represented by a node, shown with bold outlines, and every node is associated with a *control* (either PC, U, R1, R2). Controls represent the kinds of nodes, and have fixed *arities* that determine their number of ports. Control PC marks nodes representing personal computers, and its arity is 3: in clockwise order, the ports represent a keyboard interacting with an employee U, a LAN connection interacting with another PC and open to the outside network, and the mains plug of the office R. The employee U may communicate with another one via the upper port in the picture. The nesting of nodes (place graph) is shown by the inclusion of nodes into each other; the connections (link graph) are drawn as lines.

At the top level of the nesting structure sit the *regions*. In Fig. 1 there is one sole region (the dotted box). Inside nodes there may be ‘context’ *holes*, drawn as shaded boxes, which are uniquely identified by ordinals. The hole marked by 1 represents the possibility for another user U to get into office R1 and sit in front of a PC. The hole marked by 2 represents the possibility to plug a subsystem inside office R2.

Place graphs can be seen as *arrows* over a symmetric monoidal category whose objects are finite ordinals. We write $P : m \rightarrow n$ to indicate a place graph P with m holes and n regions. In Fig. 1, the place graph of G is of type $2 \rightarrow 1$. Given the place graphs P_1, P_2, their composition $P_1 \circ P_2$ is defined only if the holes of P_1 are as many as the regions of P_2, and amounts to *filling* holes with regions, according to the number each carries. The tensor product $P_1 \otimes P_2$ is not commutative, as it lays the two place graphs one next to the other (in order), thus obtaining a graph with more regions and holes, and it ‘renumbers’ regions and holes ‘from left to right’.

Link graphs are arrows of a partial monoidal category whose objects are

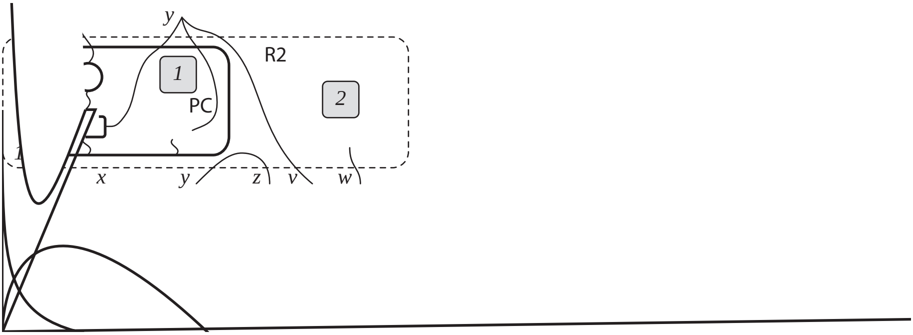

FIGURE 2. Bigraphical composition, $H \equiv G \circ (F_1 \otimes F_2)$.

(finite) sets of names. In particular, we assume a denumerable set Λ of names. A link graph is an arrow $X \to Y$, with X, Y finite subsets of Λ. The set X represents the *inner* names (drawn at the bottom of the bigraph) and Y represents the set of *outer* names (drawn on the top). The link graph connects ports to names or to *edges* (represented in Fig. 1 by a line between nodes), in any finite number. A link to a name is *open*, i.e., it may be connected to other nodes as an effect of composition. A link to an edge is *closed*, as it cannot be further connected to ports. Thus, edges are *private*, or hidden, connections. The composition of link graphs $W \circ W'$ corresponds to *linking* the inner names of W with the corresponding outer names of W' and forgetting about their identities. As a consequence, the outer names of W' (resp. inner names of W) are not necessarily inner (resp. outer) names of $W \circ W'$. Thus link graphs can perform substitution and renaming, so the outer names in W' can disappear in the outer names of this means that either names may be renamed or edges may be added to the structure. As in [16], the tensor product of link graphs is defined in the obvious way only if their inner (resp. outer) names are disjoint.

By combining ordinals with names we obtain *interfaces*, i.e., couples $\langle m, X\rangle$ where m is an ordinal and X is a finite set of names. By combining the notion of place graph and link graphs on the same nodes we obtain the notion of bigraphs, i.e., arrows $G : \langle m, X\rangle \to \langle n, Y\rangle$.

Figure 2 represents a more complex situation. Its top left-hand side reports the system of Fig. 1, in its bottom left-hand side F

names create the new links between the two structures. Intuitively, composition *first* places every region of F in the proper hole of G (place composition) and *then* joins equal inner names of G and outer names of F (link composition). In the example, as a consequence of the composition the user U in the first region of F is logged on PC, the user U in the second region of F is in room $\mathsf{R2}$. Moreover note the edge connecting the inner names y and z in G, its presence produces a link between the two users of F after the composition, imagine a phone call between the two users.

3 BiLog: syntax and semantics

The final aim of the paper is to define a logic able to describe bigraphs and their substructures. As bigraphs, place graphs, and link graphs are arrows of a (partial) monoidal category, we first introduce a meta-logical framework having monoidal categories as models; then we adapt it to model the orthogonal structures of place and link graphs. Finally, we specialise delwewe

Table 3.1. *BiLog terms*

$G, G' ::= \Omega$	constructor (for $\Omega \in \Theta$)
$G \circ G'$	vertical composition
$G \otimes G'$	horizontal composition

Table 3.2. *Typing rules*

$$\frac{type(\Omega) = I \to J}{\Omega : I \to J} \qquad \frac{G : I' \to J \quad F : I \to I'}{G \circ F : I \to J}$$

$$\frac{G : I_1 \to J_1 \quad F : I_2 \to J_2 \quad I = I_1 \otimes I_2 \quad J = J_1 \otimes J_2}{G \otimes F : I \to J}$$

Terms represent structures built on a (partial) monoid $(M, \otimes, \epsilon)$ whose elements are dubbed *interfaces* and denoted by I, J. To model nominal resources, such as heaps or link graphs, we allow the monoid to be partial.

Intuitively, terms represent typed structures with a source and a target interface ($G : I \to J$ IG □

Table 3.3. *Axioms*

Congruence Axioms:	
$G \equiv G$	Reflexivity
$G \equiv G'$ implies $G' \equiv G$	Symmetry
$G \equiv G'$ and $G' \equiv G''$ implies $G \equiv G''$	Transitivity
$G \equiv G'$ and $F \equiv F'$ implies $G \circ F \equiv G' \circ F'$	Congruence $\circ$
$G \equiv G'$ and $F \equiv F'$ implies $G \otimes F \equiv G' \otimes F'$	Congruence $\otimes$
Monoidal Category Axioms:	
$G \circ id_I \equiv G \equiv id_J \circ G$	Identity
$(G_1 \circ G_2) \circ G_3 \equiv G_1 \circ (G_2 \circ G_3)$	Associativity
$G \otimes id_\epsilon \equiv G \equiv id_\epsilon \otimes G$	Monoid Identity
$(G_1 \otimes G_2) \otimes G_3 \equiv G_1 \otimes (G_2 \otimes G_3)$	Monoid Associativity
$id_I \otimes id_J \equiv id_{I \otimes J}$	Interface Identity
$(G_1 \otimes F_1) \circ (G_2 \otimes F_2) \equiv (G_1 \circ G_2) \otimes (F_1 \circ F_2)$	Bifunctoriality

terms in general. When the framework is instantiated, terms specialise to represent particular structures and the logic specialises to describe such a particular structures as well. The semantics of a BiLog formula corresponds to a sets of terms. The logic will feature spatial connectives in the sense Spatial Logics [1, 7].

3.2 Transparency

In general not every structure of the model corresponds to an observable structure in a spatial logic. A classical example is ambient logic. Some mobile ambient constructors have their logical equivalent, e.g. ambients, and other ones are not directly mapped in the logic, e.g. the **in** and **out** prefixes. In this case the observability of the structure is distinguished from the observability of the computational terms: some terms are used to express behaviour and other to express structure. Moreover there are terms representing both structure and possible behaviour, since ambients can be opened.

The structure may be used not only to represent the distribution or the shape of resources but also to encode their behaviour. We may want to avoid a direct representation of some structures at logical level of BiLog. A natural solution is to define a notion of *transparency* over the structure. In such a way, entities really representing the structure are *transparent*, while entities encoding behaviour are *opaque* and cannot be distinguished by the logical spatial connectives. As bifunctorial terms are interpreted as arrows, transparent terms allow the logic to see their entire structure till the source interface, while opaque terms block the inspection at some middle point. A notion of transparency can also appear in models without temporal behaviour. In fact, consider a model with an access control policy defined on the structure. The policy may be variable and defined on constructors by the administrator. Thus, some terms may be transparent or opaque, depending on the current policy, and the visibility in the logic, or in the query language, will be influenced by this.

When the model is dynamic, the reacting contexts, namely those with a possible temporal evolution, are specified with an activeness predicate. We may be tempted to identify transparency as the activeness of terms. Although these concepts coincide in some case, in general they are completely orthogonal. There may be transparent terms that are active, such as a public location/directory; opaque terms that are active, such as an agent that hides its content; passive transparent terms, such as a script code; and passive opaque terms, such as controls encoding synchronisation. Indeed, the transparency is *orthogonal* to the concept of activeness.

More generally the transparency predicate avoids that every single term in the structure is mapped to its logical equivalent. Models can have additional structure not observable. Consider, as another example, an XML document. We may want to consider theCm8W .uiv9:aa au3ee-251nothd(noes;244(thr)-25040)15(xample
si25(aleonal)]T73(poli5(alw)-240(eqee-2571)t(h)-e-2571)gical3207inthatadNo

Table 3.4. *BiLog(M,* □, □, Θ, ≡, τ*)*

$\mathbf{\Omega}$ ::= $\mathbf{id}_I$ | ...	a constant formula for every Ω s.t. τ(Ω)		
A, B ::= $\mathbf{F}$	false	$A \square B$	implication
$\mathbf{id}$	identity	$\mathbf{\Omega}$	constant constructor
$A \square B$	tensor product	$A \circ B$	composition
$A \square B$	left comp. adjunct	$A \square B$	right comp. adjunct
$A \square\!- B$	left prod. adjunct	$A -\!\square B$	right prod. adjunct

$G \models \mathbf{F}$	iff	never
$G \models A \square B$	iff	$G \models A$ implies $G \models B$
$G \models \mathbf{\Omega}$	iff	$G \equiv \Omega$
$G \models \mathbf{id}$	iff	exists I s.t. $G \equiv id_I$
$G \models A \square B$	iff	exists G_1, G_2 s.t. $G \equiv G_1 \square G_2$, with $G_1 \models A$ and $G_2 \models B$
$G \models A \circ B$	iff	exists G_1, G_2. s.t. $G \equiv G_1 \circ G_2$, with τ(G_1) and $G_1 \models A$ and $G_2 \models B$
$G \models A \square B$	iff	for all $G^{\square}$, the fact that $G^{\square} \models A$ and τ($G^{\square}$) and ($G^{\square} \circ G$)↓ implies $G^{\square} \circ G \models B$
$G \models A \square B$	iff	τ(G) implies that for all $G^{\square}$, if $G^{\square} \models A$ and ($G \circ G^{\square}$)↓ then $G \circ G^{\square} \models B$
$G \models A \square\!- B$	iff	for all $G^{\square}$, the fact that $G^{\square} \models A$ and ($G^{\square} \square G$)↓ implies $G^{\square} \square G \models B$
$G \models A -\!\square B$	iff	for all $G^{\square}$, the fact that $G^{\square} \models A$ and ($G \square G^{\square}$)↓ implies $G \square G^{\square} \models B$

see that when all terms are observable the logical equivalence corresponds to ≡. Otherwise, it can be less discriminating. We assume that id_I .425 m( )]TJ/F88 and place graph. The *vertical decomposition* formula $A \circ B$ is satisfied by terms that can be the composition of terms satisfying A and B. We shall see that in some cases both the connectives correspond to well known spatial connectives. We define the *left* and *right adjuncts* for composition and tensor to express extensional properties. The left adjunct $A \circ\!- B$ expresses the property of a term to satisfy B whenever inserted in a context satisfying A. Similarly, the right adjunct $A -\!\circ B$ expresses the property of a context to satisfy B whenever filled with a term satisfying A. A similar description for $\otimes-$ and $-\otimes$, the adjoints of $\otimes$. They collapse if the tensor is commutative in the model.

3.4 Properties

Here we show some basic results about BiLog. In particular, we observe that, in presence of trivial transparency, the induced logical equivalence coincides with the structural congruence of the terms. Such a property is fundamental to describe, query and reason about bigraphical data structures, as e.g. XML (cf. [12]). In other terms, BiLog is *intensional* in the sense of [25], namely it can observe internal structures, as opposed to the extensional logics used to observe the behaviour of dynamic system. Inspired by [15], it would be possible to study a fragment of BiLog without the intensional operators $\otimes$, $\circ$, and constants.

The lemma below states that the relation $\models$ respects the congruence.

Lemma 1 (Congruence preservation). *For every couple of term G and G':*

$$\textit{if} \quad G \models A \textit{ and } G \equiv G' \quad \textit{then} \quad G' \models A.$$

Proof. Induction on the structure of the formula, by recalling that the congruence is required to preserve the typing and the transparency. In detail

Case **F**. Nothing to prove.

Case $\mathbf{\Omega}$. By hypothesis $G \models \mathbf{\Omega}$ and $G \equiv G'$. By definition $G \equiv \Omega$ and by transitivity $G' \equiv \Omega$, thus $G' \models \mathbf{\Omega}$.

Case **id**. By hypothesis $G \models \mathbf{id}$ and $G \equiv G'$. Hence there exists an I such that $G' \equiv G \equiv id_I$ and so $G' \models \mathbf{id}$.

Case $A \Rightarrow B$. By hypothesis $G \models A \Rightarrow B$ and $G \equiv G'$. This means that if $G \models A$ then $G \models B$. By induction if $G' \models A$ then $G \models A$. Thus if $G' \models A$ then $G \models B$ and again by induction $G' \models B$.

Case $A \otimes B$. By hypothesis $G \models A \otimes B$ and $G \equiv G'$. Thus there exist G_1, G_2B

C□□□ $A \square B$. By hypothesis $G \models A \square B$ and $G \equiv G^{\square}$. Thus for every $G^{\square\square}$ such that $G^{\square\square} \models A$ and $\tau(G^{\square\square})$ and $(G^{\square\square} \circ G){\downarrow}$ it holds $G^{\square\square} \circ G \models B$. Now $G \equiv G^{\square}$ implies $G^{\square\square} \circ G \equiv G^{\square\square} \circ G^{\square}$; moreover the congruence preserves typing, so $(G^{\square\square} \circ G^{\square}){\downarrow}$. By induction $G^{\square\square} \circ G^{\square} \models B$, then conclude $G^{\square} \models A \square B$.

C□□□ $A \square B$. If $\tau(G^{\square})$ is not verified, then $G^{\square} \models A \square B$ trivially holds. Suppose $\tau(G^{\square})$ to be verified. As $G \equiv G^{\square}$ and transparency preserves congruence, $\tau(G)$ is verified as well. By hypothesis for each $G^{\square\square}$ satisfying A such that $(G \circ G^{\square\square}){\downarrow}$ it holds $G \circ G^{\square\square} \models B$, and by induction $G^{\square} \circ G^{\square\square} \models B$, as $G \equiv G^{\square}$ and $(G \circ G^{\square\square}){\downarrow}$ implies $(G^{\square} \circ G^{\square\square}){\downarrow}$ and $G \circ G^{\square\square} \equiv G^{\square} \circ G^{\square\square}$. This proves $G^{\square} \models A \square B$.

C□□□ $A \square{-} B$ (and symmetrically $A -\square B$). By hypothesis $G \models A \square{-} B$ and $G \equiv G^{\square}$. Thus for each $G^{\square\square}$ such that $G^{\square\square} \models A$ and $(G^{\square\square} \square G){\downarrow}$ then $G^{\square\square} \square G \models B$. Now $G \equiv G^{\square}$ implies $G^{\square\square} \square G \equiv G^{\square\square} \square G^{\square}$, again the congruence must preserve typing so $(G^{\square\square} \square G^{\square}){\downarrow}$. Thus by induction $G^{\square\square} \square G^{\square} \models B$. The generality of $G^{\square\square}$ implies $G^{\square} \models A \square{-} B$.

□

BiLog induces a logical equivalence $=_L$ on terms in the usual sense. We say that $G_1 =_L G_2$ if for every formula A, $G_1 \models A$ implies $G_2 \models A$ and vice versa. It is easy to prove that the logical equivalence corresponds to the congruence in the model if the transparency predicate is totally verified.

Theorem 1 (Logical equivalence and congruence). *If the transparency predicate is verified on every term, then for every term G, $G^{\square}$ it holds $G =_L G^{\square}$ if and only if $G \equiv G^{\square}$.*

Proof. The forward direction is proved by defining the characteristic formula for terms, as every term can be expressed as a formula. In fact, the transparency predicate is total, hence every constant term corresponds to a toand oL.constant0

4 BiLog: derived operators

Table 4.1 outlines some interesting operators that can be derived in BiLog. The classical operators and those constraining the interfaces are self-explanatory. The ‘dual’ operators need a few explanations. The formula A □ B is satisfied by terms G such that for every possible decomposition $G \equiv G_1$ □ G_2 either $G_1 \models A$ or $G_2 \models B$. For instance, A □ A describes terms where A is true in, at least, one part of each □-decomposition. The formula $\mathbf{F}$ □ $(\mathbf{T}_{\to I}$ □ $A)$ □ $\mathbf{F}$ describes those terms where every component with outerface I satisfies A. Similarly, the composition $A \bullet B$ expresses structural properties universally quantified on every $\circ$-decomposition. Both these connectives are useful to specify security properties or types.

The adjunct dual A □ B describes terms that can be inserted into a particular context satisfying A to obtain a term satisfying B, it is a sort of existential quantification on contexts. For instance $(\mathbf{\Omega}_1$ □ $\mathbf{\Omega}_2)$ □ A describes the union between the class of two-region bigraphs (with no names in the outerface) whose merging satisfies A, and terms that can be inserted either in Ω_1 or Ω_2 resulting in a term satisfying A. Similarly the dual adjunct A □ B describes contextual terms G such that there exists a term satisfying A that inserted in G gives a term satisfying B.

The formulae $A^{□□}$, $A^{□□}$, $A^{□\circ}$, and $A^{□\circ}$ correspond to quantifications on the horizontal/vertical structure of terms. For instance $\mathbf{\Omega}^{□\circ}$ describes terms that are a finite (possibly empty) composition of simple terms Ω. The two last spatial modalities are discussed in the next section.

A first property involving the derived connectives is stated in the following lemma, proving that the interfaces for transparent terms can be observed.

Lemma 2 (Type observation). *For every term G, it holds: $G \models A_{I \to J}$ if and only if $G : I \to J$ and $G \models A$ and* $\tau(G)$*.*

Proof. For the forward direction, assume that $G \models A_{I \to J}$, then $G \equiv id_J \circ G^{□} \circ id_I$ with $G^{□} \models A$ and $\tau(G^{□})$. Now, $id_J \circ G^{□} \circ id_I : I \to J$

Table 4.1. *Derived Operators*

$\mathbf{T}$, □, □, ⇔, □, ¬			Classical operators
A_I	$\overset{def}{=}$	$A \circ \mathbf{id}_I$	Constraining the source to be I
$A_{\to J}$	$\overset{def}{=}$	$\mathbf{id}_J \circ A$	Constraining the target to be J
$A_{I \to J}$	$\overset{def}{=}$	$(A_I)_{\to J}$	Constraining the type to be $I \to J$
$A \circ_I B$	$\overset{def}{=}$	$A \circ \mathbf{id}_I \circ B$	Composition with interface I
A □$_J$ B	$\overset{def}{=}$	$A_{\to J}$ □ B	Contexts with J as target guarantee
A □$_I$ B	$\overset{def}{=}$	A_I □ B	Composing with terms having I as source
A □ B	$\overset{def}{=}$	$\neg(\neg A$ □ $\neg B)$	Dual of tensor product
$A \bullet B$	$\overset{def}{=}$	$\neg(\neg A \circ \neg B)$	Dual of composition
A □ B	$\overset{def}{=}$	$\neg(\neg A$ □ $\neg B)$	Dual of composition left adjunct
A □ B	$\overset{def}{=}$	$\neg(\neg A$ □ $\neg B)$	Dual of composition right adjunct
A^□□	$\overset{def}{=}$	$\mathbf{T}$ □ A □ $\mathbf{T}$	Some horizontal term satisfies A
A^□□	$\overset{def}{=}$	$\mathbf{F}$ □ A □ $\mathbf{F}$	Every horizontal term satisfies A
A^□∘	$\overset{def}{=}$	$\mathbf{T} \circ A \circ \mathbf{T}$	Some vertical term satisfies A
A^□∘	$\overset{def}{=}$	$\mathbf{F}$ ❢*Tdef!*	

$\overset{def}{=} \circ A$

Proposition 1. *For every term G of type □ → J, it is the case that*

$$G \models \Diamond A \text{ if and only if there exists } G^{□} □ G \text{ such that } G^{□} \models A.$$

Proof.

valid when $(I \otimes J)\downarrow$.

In general, given two formulae A, B we say that A *yields* B, and we write $A \vdash B$, if for every term G it is the case that $G \models A$ implies $G \models B$. Moreover, we write $A \dashv\vdash B$ to say both $A \vdash B$ and $B \vdash A$.

Assume that I and J are two interfaces such that their tensor product $I \otimes J$ is defined. Then, the bifuctoriality property in the logic is expressed by

$$(A_I \circ B_{\to I}) \otimes (A'_J \circ B'_{\to J}) \dashv\vdash (A_I \otimes A'_J) \circ (B_{\to I} \otimes B'_{\to J}). \tag{1}$$

In fact, we prove the following

Proposition 2. *Whenever* $(I \otimes J)\downarrow$ *, the equation (1) holds in the logic.*

Proof. Prove separately the two way of the satisfaction. First prove

$$(A_I \circ B_{\to I}) \otimes (A'_J \circ B'_{\to J}) \vdash (A_I \otimes A'_J) \circ (B_{\to I} \otimes B'_{\to J})$$

Assume that $G \models (A_I \circ B_{\to I}) \otimes (A'_J \circ B'_{\to J})$. This means that there exist $G' : I' \to I''$, $G'' : J' \to J''$ such that $I' \otimes J'$ and $I'' \otimes J''$ are defined, and $G \equiv G' \otimes G''$, with $G' \models A_I \circ B_{\to I}$ and $G'' \models A'_J \circ B'_{\to J}$. Now, $G' \models A_I \circ B_{\to I}$ means that there exist G_1 and G_2 such that $G' \equiv G_1 \circ G_2$ and

- $G_1 : I \to J'$, with $\tau(G_1)$ and $G_1 \models A$
- $G_2 : I' \to I$, with $G_2 \models B$

Similarly, $G'' \models A'_J \circ B'_{\to J}$ means $G'' \equiv G'_1 \circ G'_2$ and

- $G'_1 : J \to J''$, with $\tau(G'_1)$ and $G'_1 \models A'$
- $G'_2 : I'' \to J$, with $G_2 \models B'$

In particular, conclude $G \equiv (G_1 \circ G_2) \otimes (G'_1 \circ G'_2)$. As $I \otimes J$ is defined, $(G_1 \otimes G'_1) \circ (G_2 \otimes G'_2)$ is an admissible composition. The bifunctoriality property implies $G \equiv (G_1 \otimes G'_1) \circ (G_2 \otimes G'_2)$. Moreover $\tau(G_1 \otimes G'_1)$, as $\tau(G_1)$ and $\tau(G'_1)$. Hence conclude that $G \models (A_I \otimes A'_J) \circ (B_{\to I} \otimes B'_{\to J})$, as required.

For the converse, prove

$$(A_I \otimes A'_J) \circ (B_{\to I} \otimes B'_{\to J}) \vdash (A_I \circ B_{\to I}) \otimes (A'_J \circ B'_{\to J}).$$

Assume that $G \models (A_I \otimes A'_J) \circ (B_{\to I} \otimes B'_{\to J})$. By following the same lines as before, deduce that $G \equiv (G_1 \otimes G'_1) \circ (G_2 \otimes G'_2)$, where

- $\tau(G_1 \otimes G'_1)$
- $G_1 : I \to J'$ such that $G_1 \models A$
- $G'_1 : J \to J''$ such that $G'_1 \models A'$
- $G_2 : I' \to I$ such that $G_2 \models B$
- $G'_2 : I'' \to J$ such that $G_2 \models B'$

Also in this case, we the tensor product of the required interfaces can be performed. Hence compose $(G_1 \circ G_2) \otimes (G'_1 \circ G'_2)$. Again, the bifunctoriality property implies $G \equiv (G_1$

Table 5.1. *Additional Axioms for Place Graphs Structural Congruence*

Table 5.2. *Information tree Terms (over Λ) and congruence*

$T, T^{\square} ::=$	0	empty tree consisting of a single root node
	$a[T]$	single edge tree labelled $l \square$ Λ leading to the subtree T
	$T \mid T^{\square}$	tree obtained by merging the roots of the trees T and $T^{\square}$

$T \mid 0 \equiv T$	neutral element
$T \mid T^{\square} \equiv T^{\square} \mid T$	commutativity
$(T \mid T^{\square}) \mid T^{\square\square} \equiv T \mid (T^{\square} \mid T^{\square\square})$	associativity

Table 5.3. *Propositional Spatial Tree Logic*

$A, B ::=$	$\mathbf{F}$	anything	$a[A]$	location
	$\mathbf{0}$	empty tree	$A@a$	location adjunct
	$A \;\square\; B$	implication	$A \mid B$	composition
			$A \,\square\, B$	composition adjunct

$T \models_{\square\square\square} \mathbf{F}$	iff	never
$T \models_{\square\square\square} \mathbf{0}$	iff	$F \equiv 0$
$T \models_{\square\square\square} A \;\square\; B$	iff	$T \models_{\square\square\square} A$ implies $T \models_{\square\square\square} B$
$T \models_{\square\square\square} a[A]$	iff	there exists $T^{\square}$ s.t. $T \equiv a[F^{\square}]$ and $T^{\square} \models_{\square\square\square} A$
$T \models_{\square\square\square} A@a$	iff	$a[T] \models_{\square\square\square} A$
$T \models_{\square\square\square} A \mid B$	iff	there exists T_1, T *and*

Table 5.4. *Encoding STL in PGL over prime ground place graphs*

Trees into Prime Ground Place Graphs

$[\![\, 0 \,]\!] \overset{def}{=} \mathbf{1}$ $\quad$ $[\![\, a[T] \,]\!] \overset{def}{=} \mathsf{K}(a) \circ [\![\, T \,]\!]$ $\quad$ $[\![\, T_1 \mid T_2 \,]\!] \overset{def}{=} join \circ ([\![\, T_1 \,]\!] \square [\![\, T_2 \,]\!])$

STL formulae into PGL formulae

$[\![\, \mathbf{0} \,]\!] \overset{def}{=} 1$ $\qquad$ $[\![\, a[A] \,]\!] \overset{def}{=} \mathsf{K}(a) \circ_1 [\![\, A \,]\!]$

$[\![\, \mathbf{F} \,]\!] \overset{def}{=} \mathbf{F}$ $\qquad$ $[\![\, A@a \,]\!] \overset{def}{=} \mathsf{K}(a) \;\square_1 [\![\, A \,]\!]$

$[\![\, A \;\square\; B \,]\!] \overset{def}{=} [\![\, A \,]\!] \;\square\; [\![\, B \,]\!]$ $\qquad$ $[\![\, A \mid B \,]\!] \overset{def}{=} [\![\, A \,]\!] \mid [\![\, B \,]\!]$

$[\![\, A \square B \,]\!] \overset{def}{=} ([\![\, A \,]\!] \mid \mathbf{id}_1) \;\square_1 [\![\, B \,]\!]$

we remark that: *(i)* the parallel composition of STL is the structural commutative separation of PGL; *(ii)* tree labels can be represented by the corresponding controls of the place graph; *(iii)* location and composition adjuncts of STL are enc$\overset{def}{o}$ded by the left composition adjunct, as they add logically expres0 the]]] it is easy to see that the encodings $[\![\ ]\!]$ and $(\![\ ]\!)$ are one the inverse of the other, hence they give a bijection from trees to prime ground place graphs, fundamental in the proof of the following theorem.

Theorem 2 (Encoding STL). *For each tree T and formula A of STL:*

$$T \models_{\square\square\square} A \quad \textit{if and only if} \quad [\![T]\!] \models [\![A]\!].$$

Proof. The theorem is proved by structural induction on STL formulae. The transparency predicate is not considered here, as it is verified on every control. The basic step deals with the constants **F** and **0**. Case **F** follows by definition. For the case **0**, $[\![T]\!] \models [\![\mathbf{0}]\!]$ means $[\![T]\!] \models 1$, that by definition is $[\![T]\!] \equiv 1$ and so $T \equiv (\![[\![T]\!]]\!) \equiv (\![1]\!)$

case that for every $g : 0 \rightarrow 1$ such that $g \models [\![A]\!]$ it holds $join(g \,\square\, id_1) \circ [\![T]\!] \models [\![B]\!]$, that is $join(g \,\square\, [\![T]\!]) \models [\![B]\!]$ by bifunctoriality property. Since the encoding is a bijection, this is equivalent to say that for every tree $T^{\square}$ such that $[\![T^{\square}]\!] \models [\![A]\!]$ it holds $join([\![T^{\square}]\!] \,\square\, [\![T]\!]) \models [\![B]\!]$, that is $[\![T^{\square} \mid T]\!] \models [\![B]\!]$. By induction hypothesis, for every $T^{\square}$ such that $T^{\square} \models_{\square\square\square} A$ it holds $T^{\square} \mid T \models$

Table 5.5. *Additional Axioms for Link Graph Structural Congruence*

Link Axioms:	
$^{a}/_{a} \equiv id_{a}$	Link Identity
$/a \circ {}^{a}/_{b} \equiv /b$	Closing renaming
$/a \circ a \equiv id_{□}$	Idle edge
$^{b}/_{(Y□a)} \circ (id_{Y} □ {}^{a}/_{X}) \equiv {}^{b}/_{Y□X}$	Composing substitutions
Link Node Axiom:	
$\alpha \circ \mathsf{K}_{\vec{a}} \equiv \mathsf{K}_{\alpha(\vec{a})}$	Renaming

and $k = ar(\mathsf{K})$. The control $\mathsf{K}_{\vec{a}}$ represents a resource of kind K with named ports $\vec{a}$. Any ports may be connected to other node ports via wiring compositions.

In this case, the structural congruence $\equiv$ is refined as outlined in Tab. 5.5 with obvious axioms for links, modelling α-conversion and extrusion of closed names. We assume the transparency predicate τ verified for wiring constructors.

Fixed the transparency predicate τ for each control in K, the Link Graph Logic LGL(K, τ) is $BiLog(\mathsf{P}_{fin}(\Lambda), □, □, \equiv, \mathsf{K} □ \{/a, {}^{a}/_{X}$

$[W']W$ for $(W' \otimes id_{X'\setminus Y}) \circ W$ and if $\vec{a} = a_1, \ldots, a_n$ and $\vec{b} = b_1, \ldots, b_n$, we write $\vec{a} \leftarrow \vec{b}$ for $a_1 \leftarrow b_1 \otimes \ldots \otimes a_n \leftarrow b_n$, similarly for $\vec{a} \Leftrightarrow \vec{b}$. From the tensor product it is possible to derive a product with sharing on $\vec{a}$. Given $G : X \to Y$ and $G' : X' \to Y'$ with $X \cap X' = \emptyset$, we choose a list $\vec{b}$ (with the same length as $\vec{a}$) of fresh names. The composition with sharing $\vec{a}$ is

$$G \overset{\vec{a}}{\otimes} G' \overset{def}{=} [\vec{a} \Leftrightarrow \vec{b}](([\vec{b} \leftarrow \vec{a}] \circ G) \otimes G').$$

In this case, the tensor product is well defined since all the common names $\vec{a}$ in W are renamed to fresh names, while the sharing is re-established afterwards by linking the $\vec{a}$ names with the $\vec{b}$ names.

By extending this sharing to all names we define the parallel composition $G \mid G'$ as a total operation. However, such an operator does not behave ‘well’ with respect to the composition, as shown in [19]. In addition a direct inclusion of a corresponding connective in the logic would impact the satisfaction relation by expanding the finite horizontal decompositions to the boundless possible name-sharing decompositions. (This may be the main reason why logics describing models with name closure and parallel composition are undecidable [11].) This is due to the fact that the set of names shared by a parallel composition is not known in advance, and therefore parallel composition can only be defined by using an existential quantification over the entire set of shared names.

Names can be internalised and effectively made private to a bigraph by the closure operator $/a$. The effect of composition with $/a$ is to add a new edge with no public name, and therefore to make a to 8 9.963 Tf 5lofe

n1e. -2.h33Td[(using)-250(an)-250(e)15(xistent-250(e)15(xistential)-55 Td[it nt-7(Tn3a(shl3-28

Table 5.6. *Spatial graph Terms (with local names) and congruence*

$G, G^{□}$ ::=	nil	empty graph
	$a(x, y)$	single edge graph labelled a □ Λ connecting the nodes x, y
	$G \mid G^{□}$	composing the graphs $G, G^{□}$, with sharing of nodes
	$(\nu x)G$	the node x is local in G

$G \mid nil \equiv G$	neutral element
$G \mid G^{□} \equiv G^{□} \mid G$	commutativity
$(G \mid G^{□}) \mid G^{□□} \equiv G \mid (G^{□} \mid G^{□□})$	associativity
y □ $fn(G)$ implies $(\nu x)G \equiv (\nu y)G\{x \leftarrow y\}$	renaming
$(\nu x)nil \equiv nil$	extrusion Zero
x □ $fn(G)$ implies $G \mid (\nu x)G^{□} \equiv (\nu x)(G \mid G^{□})$	extrusion composition
x □ y, z implies $(\nu x)a(y, z) \equiv a(y, z)$	extrusion edge

(ν*x*)f 3.318 0 Td[(∞)]J/F88 9o 3.367Ze(367Ze(3)]T0 Td[(j)]51 0 Td[(z)]TJ/F78 9.963

Table 0]TJ/F88 9.963 Tf 42.411 0 Td[(Spatial)-250(gr)15(aph)-250(T)92(erms)-050with local n

Table 5.7. *Propositional Spatial Graph Logic (SGL)*

$\phi, \psi ::=$	$\mathbf{F}$	false	$a(x,y)$	an edge from x to y
	$\mathbf{nil}$	empty graph	$\phi \mid \psi$	composition
	$\phi \Rightarrow \psi$	implication		

$G \models_{SGL} \mathbf{F}$	iff	never
$G \models_{SGL} \mathbf{nil}$	iff	$G \equiv nil$
$G \models_{SGL} \phi \Rightarrow \psi$	iff	$G \models_{SGL} \phi$ implies $G \models_{SGL} \psi$
$G \models_{SGL} a(x,y)$	iff	$G \equiv a(x,y)$
$G \models_{SGL} \phi \mid \psi$	iff	there exists G_1, G_2 s.t. $G \equiv G_1 \mid G_2$ and $G_1 \models_{SGL} \phi$ and $G_2 \models_{SGL} \psi$

Table 5.8. *Encoding Propositional SGL in LGL over ground link graphs*

Spatial Graphs into Two-ported Ground Link Graphs

$$[\![nil]\!]_X \stackrel{def}{=} X$$

$$[\![a(x,y)]\!]_X \stackrel{def}{=} \mathsf{K}(a)_{x,y} \otimes X \setminus \{x,y\}$$

$$[\![(\nu x)G]\!]_X \stackrel{def}{=} ((/x \otimes id_{X\setminus\{x\}}) \circ [\![G]\!]_{\{x\}\uplus X})) \otimes (\{x\} \cap X)$$

$$[\![G \mid G']\!]_X \stackrel{def}{=} [\![G]\!]_X \overset{X}{\otimes} [\![G']\!]_X$$

SGL formulae into LGL formulae

$$[\![\mathbf{nil}]\!]_X \stackrel{def}{=} X \qquad [\![a(x,y)]\!]_X \stackrel{def}{=} \mathsf{K}(a)_{x,y} \otimes (X \setminus \{x,y\})$$

$$[\![\mathbf{F}]\!]_X \stackrel{def}{=} \mathbf{F} \qquad [\![\phi \Rightarrow \psi]\!]_X \stackrel{def}{=} [\![\phi]\!]_X \Rightarrow [\![\psi]\!]_X$$

$$[\![\phi \mid \psi]\!]_X \stackrel{def}{=} [\![\phi]\!]_X \overset{X}{\otimes} [\![\psi]\!]_X$$

type $\epsilon \to \langle 1, X \rangle$. The results in [19] say that a bigraph without nested nodes and $\langle 1, X \rangle$ as outerface have the following normal form (where $Y \subseteq X$):

$$G ::= (/Z \mid id_{\langle 1,X \rangle}) \circ (X \mid M_0 \mid \ldots \mid M_{k-1})$$
$$M ::= \mathsf{K}_{x,y}(a) \circ 1$$

The inverse encoding is based on such a normal form:

$$(\![(/Z \mid id_{\langle 1,X \rangle}) \circ (X \mid M_0 \mid \ldots \mid M_{k-1})]\!) \stackrel{def}{=} (\nu Z)\,(nil \mid (\![M_0]\!) \mid \ldots \mid (\![M_{k-1}]\!))$$
$$(\![\mathsf{K}_{x,y}(a) \circ 1]\!) \stackrel{def}{=} a(x,y)$$

Notice that the extrusion properties of local names correspond to node and link axioms. The encodings $[\![\ ]\!]$ and $(\![\ ]\!)$ provide a bijection, up to congruence, between graphs of SGL and ground link graphs with outer face X and built by controls of arity 2. ■

The previous lemma is fundamental in proving that the soundness of the encoding for SGL in BiLog, stated in the following theorem.

Theorem 3 (Encoding SGL). *For every graph G, every finite set X containing $fn(G)$, and every formula ϕ of the propositional fragment of SGL:*

$$G \models_{\text{□□□}} \phi \quad \textit{if and only if} \quad [\![G]\!]_X \models [\![\phi]\!]_X .$$

Proof. By induction on formulae of SGL. The transparency predicate is not considered here, as it is verified on every control. The basic step deals with the constants **F**, **nil** and a

Table 5.9. *Additional axioms for Bigraph Structural Congruence*

Symmetric Category Axioms:	
$\gamma_{I,\square} \equiv id_I$	Symmetry Id
$\gamma_{I,J} \circ \gamma_{J I}$*gruence*	

ity and connectivity. To testify this, §5.7 shows how recently proposed Context Logic for Trees (CTL) [4] can be encoded into bigraphs. The idea of the encoding is to extend the encoding of STL with (single-hole) contexts and identified nodes. First, §5.6 gives some details on the transparency predicate.

5.6 Transparency on bigraphs

In the logical framework we gave the minimal restrictions on the transparency predicate to prove our results. Here we show a way to define a transparency predicate. The most natural way is to make the transparent terms a sub-category of the more general category of terms. This essentially means to impose the product and the composition of two transparent terms to be transparent.

Thus transparency on all terms is derived from a transparency policy $\tau_\Theta(\,)$ defined only on the constructors. Note that the transparency definition depends also on the congruence. In the following definition we show how to derive the transparency from a transparency policy.

Definition 2 (Transparency). *Given the monoid of interfaces* $(M, \otimes, \epsilon)$*, the set of constructors* Θ*, the congruence* $\equiv$ *and a transparency policy predicate* τ_Θ *defined on the constructors in* Θ *we define the transparency on terms as follows:*

$$\frac{G \equiv id_I}{\tau(G)} \qquad \frac{\exists I.G : \epsilon \to I}{\tau(G)} \qquad \frac{G \equiv \Omega \quad \tau_\Theta(\Omega)}{\tau(G)}$$

$$\frac{G \equiv G_1 \otimes G_2 \quad \tau(G_1) \quad \tau(G_2)}{\tau(G)} \qquad \frac{G \equiv G_1 \circ G_2 \quad \tau(G_1) \quad \tau(G_2)}{\tau(G)}$$

Next lemma proves that the condition we posed on the transparency predicate holds for this particular definition.

Lemma 5 (Transparency properties). *If G is ground or G is an identity then* $\tau(G)$ *is verified. Moreover, if* $G \equiv G'$ *then* $\tau(G)$ *is equivalent to* $\tau(G')$.

Proof. The former statement is verified by definition. The latter is proved by induction on the derivations. □

We assume every bigraphical constructor, that is not a control, to be transparent and the transparency policy to be defined only on the controls. The transparency the policy can be defined. for instance, for security reasons.

5.7 Encoding CTL

Paper [4] presents a spatial context logic to describe programs manipulating a tree structured memory. The model of the logic is the set of unordered labelled trees T and *linear contexts* C, which are trees with a unique hole. Every node has a name, so to identify memory locations. From the model, the logic is dubbed Context Tree Logic, CTL in the following. Given a denumerable set of labels and a denumerable set of identifiers, trees and contexts are defined in Tab. 5.10:

Table 5.11. *Context Tree Logic (CTL)*

$P, P^{\square}$ ::=	*false*	
	0	empty tree formula
	$K(P)$	context application
	$K \square P$	context application adjunct
	$P \square P^{\square}$	implication
$K, K^{\square}$::=	*false*	
	$-$	identity context formula
	$a_x[K]$	node context formula
	$P \square P^{\square}$	context application adjunct
	$P \mid K$	parallel context formula
	$K \square K^{\square}$	implication

Table 5.12. *Semantics for CTL*

$T \models_{\mathsf{T}} false$	iff	never
$T \models_{\mathsf{T}} \mathbf{0}$	iff	$T \equiv 0$
$T \models_{\mathsf{T}} K(P)$	iff	there exist $C, T^{\square}$ s.t. $C(T^{\square})$ well-formed, and $T \equiv C(T^{\square})$ and $C \models_{\mathsf{K}} K$ and $T^{\square} \models_{\mathsf{T}} P$
$T \models_{\mathsf{T}} K \square P$	iff	for every C: $C \models_{\mathsf{K}} K$ and $C(T)$ well-formed implies $C(T) \models_{\mathsf{T}} P$
$T \models_{\mathsf{T}} P \square P^{\square}$	iff	$T \models_{\mathsf{T}} P$ implies $T \models_{\mathsf{T}} P^{\square}$
$C \models_{\mathsf{K}} false$	iff	never
$C \models_{\mathsf{K}} -$	iff	$C \equiv -$
$C \models_{\mathsf{K}} a_x[K]$	iff	there exists $C^{\square}$ s.t. $a_x[C^{\square}]$ well-formed, and $C \equiv a_x[C^{\square}]$ and $C^{\square} \models_{\mathsf{K}} K$
$C \models_{\mathsf{K}} P \square P^{\square}$	iff	for every T: $T \models_{\mathsf{T}} P$ and $C(T)$ well-formed implies $C(T) \models_{\mathsf{T}} P^{\square}$
$C \models_{\mathsf{K}} P \mid K$	iff	there exist $C^{\square}, T$ s.t. $T \mid C^{\square}$ well-formed, and $C \equiv T \mid C^{\square}$ and $T \models_{\mathsf{T}} P$ and $C^{\square} \models_{K} K$
$C \models_{\mathsf{K}} K \square K^{\square}$	iff	$C \models_{\mathsf{K}} K$ implies $T \models_{\mathsf{T}} K^{\square}$

formula $\mathbf{id}_{\langle m,\_\rangle}$ to represent identities on places by constraining the place part of the interface to be fixed and leaving the name part to be free:

$$\mathbf{id}_{\langle m,\_\rangle} \stackrel{def}{=} \mathbf{id}_m \otimes (\mathbf{id} \wedge \neg(\mathbf{id}_1^{\square\square})).$$

It is easy to see that $G \models id_{\langle m,-\rangle}$ means that there exits a set of names X such that $G \equiv id_m \otimes id_X$. By using such an identity formula we define the corresponding typed composition $\circ_{\langle m,\_\rangle}$ and the typed adjuncts $\multimap_{\langle m,\_\rangle}$, $\multimapinv_{\langle m,\_\rangle}$:

$$\begin{aligned} A \circ_{\langle m,\_\rangle} B &\stackrel{def}{=} A \circ \mathbf{id}_{\langle m,\_\rangle} \circ B \\ A \multimap_{\langle m,\_\rangle} B &\stackrel{def}{=} (\mathbf{id}_{\langle m,\_\rangle} \circ A) \multimap B \\ A \multimapinv_{\langle m,\_\rangle} B &\stackrel{def}{=} (A \circ \mathbf{id}_{\langle m,\_\rangle}) \multimapinv B \end{aligned}$$

We then define the operator $|$ for the parallel composition with separation operator $\otimes$ as both a term constructor and a logical connective:

$$D \mid E \stackrel{def}{=} [join](D \otimes E) \qquad \text{for } D \text{ and } E \text{ prime bigraphs}$$

$$A \mid B \stackrel{def}{=} (\mathbf{join} \otimes \mathbf{id}_{\langle 0,\_\rangle}) \circ (A_{\to\langle 1,\_\rangle} \otimes B_{\to\langle 1,\_\rangle}) \qquad \text{for } A \text{ and } B \text{ formulae}$$

The operator $|$ enables the encoding of trees and contexts to bigraphs. In particular, we consider a signature with controls of arity 1 and we define the transparency predicate to be verified on every control. Moreover we assume a bijective function from tags to controls

$$a_x \longmapsto \mathsf{K}(a)_x.$$

The details are outlined in Tab. 5.13. The encodings of trees turn out to be *ground prime discrete bigraphs*: bigraphs with open links and type $0 \to \langle 1, X\rangle$. The result in [19] says that the normal form, up to permutations, for ground prime discrete bigraphs is:

$$g = (join_k \otimes id_X) \circ (M_1 \otimes \ldots \otimes M_k),$$

where M_i are discrete ground molecules of the form

$$M = (\mathsf{K}(a)_x \otimes id_Y)g.$$

We can now define the reverse encoding $⦇\ ⦈$ of $[\![\ ]\!]$, from ground prime discrete bigraphs to trees, involving such a normal form:

$$\begin{gathered} ⦇\, join_0 \,⦈ \stackrel{def}{=} 0 \\ ⦇\, (\mathsf{K}(a)_x \otimes id_Y) \circ g \,⦈ \stackrel{def}{=} a_x[\, ⦇\, g \,⦈\,] \\ ⦇\, (join_k \otimes id_Y) \circ (M_1 \otimes \ldots \otimes M_k) \,⦈ \stackrel{def}{=} ⦇\, M_1 \,⦈ \mid \ldots \mid ⦇\, M_k \,⦈ \end{gathered}$$

Moreover, the encodings of linear contexts turn out to be *unary discrete bigraphs G*: bigraphs with open links and type $\langle 1, X\rangle \to \langle 1, Y\rangle$. Again, the result in [19] implies that the normal form, up to permutations, for unary discrete bigraphs

Table 5.13. *Encoding CTL in BiLog over prime discrete ground bigraphs*

Trees into prime ground discrete bigraphs	Contexts into unary discrete bigraphs
	$[\![-]\!]_C \overset{def}{=} id_1$
$[\![0]\!] \overset{def}{=} 1$	$[\![a_x[C]]\!]_C \overset{def}{=} (\mathsf{K}(a)_x \square\ fn(C)) \circ [\![C]\!]_C$
$[\![a_x[T]]\!] \overset{def}{=} (\mathsf{K}(a)_x \square\ fn(T)) \circ [\![T]\!]$	$[\![T \mid C]\!]_C \overset{def}{=} [\![T]\!] \square [\![C]\!]_C$
$[\![T_1 \mid T_2]\!] \overset{def}{=} [\![T_1]\!] \square [\![T_2]\!]$	$[\![C \mid T]\!]_C \overset{def}{=} [\![C]\!]_C \square [\![T]\!]$
TL formulae into PGL formulae	CTL formulae into PGL formulae
$[\![false]\!]_P \overset{def}{=} \mathbf{F}$	$[\![false]\!]_K \overset{def}{=} \mathbf{F}$
$[\![\mathbf{0}]\!]_P \overset{def}{=} \mathbf{1}$	$[\![-]\!]_K \overset{def}{=} \mathbf{id}_1$
$[\![K(P)]\!]_P \overset{def}{=} [\![K]\!]_K \circ_{\square 1,\_\square} [\![P]\!]_P$	$[\![P \square P^{\square}]\!]_K \overset{def}{=} [\![P]\!]_P \ \square_{\square 1,\_\square} [\![P^{\square}]\!]_P$
$[\![K \square P]\!]_P \overset{def}{=} [\![K]\!]_K \ \square_{\square 1,\_\square} [\![P]\!]_P$	$[\![a_x[K]]\!]_K \overset{def}{=} ((\mathsf{K}(a)_x) \square\ id_{\square 0,\_\square}) \circ [\![K]\!]_K$
$[\![P \ \square\ P^{\square}]\!]_P \overset{def}{=} [\![P]\!]_P \ \square\ [\![P^{\square}]\!]_P$	$[\![P \mid K]\!]_K \overset{def}{=} [\![P]\!]_P \square [\![K]\!]_K$
	$[\![K \ \square\ K^{\square}]\!]_K \overset{def}{=} [\![K]\!]_K \ \square\ [\![K^{\square}]\!]_K$

is:

$$G = (join_k \square\ id_Y) \circ (R \square M_1 \square \ldots \square M_{k-1})$$

where M_i are discrete ground groR

generalisation of Context Tree Logic to contexts with several holes and regions. On the other hand, since STL is more general than separation logic, cf. [4], and it is used to characterise programs that manipulate tree structured memory model, BiLog can express separation logic as well.

6 Towards dynamics

The main aim of this paper is to introduce BiLog and its expressive power in describing static structures. BiLog is however able to deal with the dynamic behaviour of the model, as well. Essentially, this happens thanks to the contextual nature of the logic, suitable to characterise structural parametric reaction rules, expressing dynamics.

A main feature of a distributed system is mobility, or dynamics in general. In dealing with communicating and nomadic processes, the interest is not only to describe their internal structure, but also their behaviour. So far, it has been

According to the formulation of the reduction given above, we obtain

$$g \models \blacklozenge A \quad \textit{iff} \quad \textit{there exist} (R, R') \in S,\ id_Y,\ D \textit{ active, and } d \textit{ ground; such that}$$
$$g \equiv D \circ (R \otimes id_Y) \circ d\ ,\ g' \equiv D \circ (R' \otimes id_Y) \circ d \textit{ and } g' \models A. \quad (3)$$

One may wonder whether the modality ♦ is the only way to express a temporal evolution in BiLog. It turns out that BiLog has a built in notion of dynamics. In several cases, BiLog itself is sufficient to express the computation. One of them is the encoding of CCS, shown in the following.

We focus on the fairly small fragment of CCS considered in [2], consisting of prefix and parallel composition only; P, Q will range over *processes*, and $a, \overline{a}$ over actions, chosen in the enumerable set *Acts*. The syntax of the calculus is defined by the following grammar.

$$\begin{array}{lcl} P & ::== & \mathbf{0} \mid \lambda.P \mid P \,|\, P \\ \lambda & ::== & a \mid \overline{a} \end{array}$$

Note that the operator ν is not included, hence all the names appearing in a process are free, this fact yields the encoding to produce bigraphs with open links. The *structural congruence* is defined as the least congruence ≡ on processes such that $P \mid \mathbf{0} \equiv P$, $P \mid Q \equiv Q \mid P$

a finite set of names, viz., the outer names of the term that can fill the context. In particular, the controls act and coact are declared to be *passive*, i.e., no reaction can occur inside them.

As already said, we consider bigraphs built on the controls $\mathsf{act}_a, \mathsf{coact}_a$. The encoding $[\![\ ]\!]_X$ is parameterised by a *finite* subset $X \subseteq Acts$. In particular, the encoding yields ground bigraphs with outer face $\langle 1, X\rangle$ and open links. The translation for processes is formally defined as

$$
\begin{array}{rcl}
[\![\, \mathbf{0} \,]\!]_X & \stackrel{def}{=} & 1 \otimes X \\
[\![\, a.P \,]\!]_X & \stackrel{def}{=} & (\mathsf{act}_a \overset{a}{\otimes} id_X) \circ [\![\, P \,]\!]_X \\
[\![\, \overline{a}.P \,]\!]_X & \stackrel{def}{=} & (\mathsf{coact}_a \overset{a}{\otimes} id_X) \circ [\![\, P \,]\!]_X \\
[\![\, P \mid Q \,]\!]_X & \stackrel{def}{=} & join \circ ([\![\, P \,]\!]_X \overset{X}{\otimes} [\![\, Q \,]\!]_X)
\end{array}
$$

Where $a \in X$, and, with abuse of notation, the sharing/separation operator $\overset{X}{\otimes}$ stands for $\overset{\vec{a}}{\otimes}$ where $\vec{a}$ is any array of all the elements in X. Note, in particular, that the sharing tensor "$\_ \overset{a}{\otimes} id_X$" allows the process filling the hole in act_a (and coact_a) to perform other actions a. Moreover *join* makes the tensor to be commutative in the encoding of parallel, in fact there is a straight correspondence between the parallel operators in the two calculi, as $[\![\, P \mid Q \,]\!]_X$ corresponds to $[\![\, P \,]\!]_X \mid [\![\, Q \,]\!]_X$, that is the parallel operator on bigraphs. The result stated in Lemma 7 says that the encoding is bijective on prime ground bigraphs with open links. First we need a general result on bigraphs and parallel composition.

Lemma 6 (Adding Names).

2. *For every couple of processes P, Q and for every finite subset $X \subset$ Acts including the free names of P, Q it holds: $P \equiv Q$ if and only if $[\![\, P \,]\!]_X \equiv [\![\, Q \,]\!]_X$.*

Proof. Prove point (1) by showing that every prime ground bigraph with outerface $\langle 1, X\rangle$ has at least one pre-image for the translation $[\![\, \cdot \,]\!]_X$. Proceed by induction on the number of nodes in the bigraphs. First we recall the connected normal form for bigraphs. The paper [19] proves that every prime ground bigraph G with outerface $\langle 1, X\rangle$ and open links has the following Connected Normal Form:

$$\begin{aligned} G &::= X \mid F \\ F &::= M_1 \mid \ldots \mid M_k \\ M &::= (\mathsf{K}_a \mid id_Y) \circ F \qquad \text{(for } \mathsf{K}_a \in \{\mathsf{act}_a, \mathsf{coact}_a\}) \end{aligned}$$

The base of induction is the bigraph X, and clearly $[\![\, \mathbf{0} \,]\!]_X = X$. For the inductive step, consider a bigraph G with at least one node. This means $G = X \mid ((\mathsf{K}_a \mid id_Y) \circ F) \mid G'$. Without losing generality, assume $\mathsf{K}_a = \mathsf{act}_a$, so by Proposition 6:

$$G = (\mathsf{act}_a \mid id_X) \circ (X \mid F) \mid (X \mid G').$$

Now, the induction says that there exist P and Q such that $[\![\, P \,]\!]_X = X \mid F$ and $[\![\, Q \,]\!]_X$

In [22] it is proved that the translation preserves and reflects the reactions, that is: $P \longrightarrow P'$ *if and only if* $[\![P]\!] \longrightarrow [\![P']\!]$.

The reaction rules are defined as

$$(\mathsf{act}_a \mid id_{Y_1}) \mid (\mathsf{coact}_a \mid id_{Y_2}) \longrightarrow a \mid id_{\langle 1,Y_1\rangle} \mid id_{\langle 1,Y_2\rangle}.$$

This can be mildly sugared to obtain the rule introduced in (5)

Moreover, the active contexts introduced in (6) can be rephrased as

$$g \mid \square$$

where g is a single-rooted ground bigraph with open links. It is easy to conclude that the most general context ready to react has the form

$$\square_0 \mid \mathsf{act}_a \square_1 \mid \mathsf{coact}_a \square_2 \longmapsto \square_0 \mid \square_1 \mid \square_2$$

the hole $\square_0$ has to be filled in by single-rooted ground bigraphs with open links, whereas the holes $\square_1$ and $\square_2$ by ground bigraphs. Note that such a reduction is compositional with the parallel operator. In case of the CCS translation, the a reacting bigraphs are further characterised as shown in Lemma 8. In particular, the lemma shows that every reacting $[\![P]\!]_X$ can be decomposed into a redex and a bigraph with a well defined structure, that is composed with a reactum to obtain the result of the reaction. The Redex and the Reactum are formally outlined in Tab. 6.1. They will be the key point to express the next step modality in BiLog. Note that y_1 and y_2 of the definition in Tab. 6.1 have to be disjoint with X, Y_1 and Y_2. They are useful for join the action with the corresponding coaction.

Table 6.1. *Reacting Contexts for CCS*

Bigraphs:

$$Redex_a^{y_1,y_2,Y_1,Y_2} \stackrel{def}{=} W \circ (id_Y \otimes join) \circ (id_Y \otimes join \otimes id_1) \circ \{((y_1 \leftarrow a) \otimes s W = \circ ( ) \circ \{. \; y_1 \leftarrow$$

2. *There exist the bigraphs $G_1, G_2, G_3 : \square \rightarrow \square 1, X\square$ and the name $a \;\square\; X$, such that*

$$[\![\, P \,]\!]_X \equiv ((\mathsf{act}_a \mid id_X) \circ G_1) \mid ((\mathsf{coact}_a \mid id_X) \circ G_2) \mid G_3$$

   and $G \equiv G_1 \mid G_2 \mid G_3$.

3. *There exist the actions $a \;\square\; X$ and $y_1, y_2 \;\square\; X$, and two mutually disjoint subsets $Y_1, Y_2 \;\square\; Acts$ with the same cardinality as X, but disjoint with X, y_1, y_2, and there exist the bigraphs $H_1 : \square \rightarrow \square 1, Y_1\square$, $H_2 : \square \rightarrow \square 1, Y_2\square$, and $H_3 : \square \rightarrow \square 1, X\square$ with open links, such that*

$$[\![\, P \,]\!]_X \equiv Redex_a^{y_1,y_2,Y_1,Y_2} \circ (H_1 \;\square\; H_2 \;\square\; H_3)$$

   and

$$G \equiv React_a^{Y_1,Y_2} \circ (H_1 \;\square\; H_2 \;\square\; H_3),$$

   where $Redex_a^{y_1,y_2,Y_1,Y_2}$, $React_a^{Y_1,Y_2}$ are defined in Tab. 6.1.

W links y_1 and y_2 with a. By bifunctoriality property, $[\![P]\!]_X$ is rewritten as

$$\begin{aligned} &W \circ (id_Y \otimes join) \circ (id_Y \otimes join \otimes id_1) \circ \{((y_1 \leftarrow a) \otimes id_1) \circ \\ &\qquad \circ \mathsf{act}_a \otimes id_{Y_1} \otimes ((y_2 \leftarrow a) \otimes id_1) \circ \mathsf{coact}_a \otimes id_{Y_2} \otimes G_3 \} \circ \\ &\qquad\qquad \circ \{ ((Y_1 \leftarrow X) \otimes id_1) \circ G_1 \otimes ((Y_2 \leftarrow X) \otimes id_1) \circ G_2 \}, \end{aligned}$$

and, again by bifunctoriality property, as

$$\begin{aligned} &W \circ (id_Y \otimes join) \circ (id_Y \otimes join \otimes id_1) \circ \{((y_1 \leftarrow a) \otimes id_1) \circ \\ &\qquad \circ \mathsf{act}_a \otimes id_{Y_1} \otimes ((y_2 \leftarrow a) \otimes id_1) \circ \mathsf{coact}_a \otimes id_{Y_2} \otimes id_{\langle 1,X \rangle} \} \circ \\ &\qquad\qquad \circ \{ ((Y_1 \leftarrow X) \otimes id_1) \circ G_1 \otimes ((Y_2 \leftarrow X) \otimes id_1) \circ G_2 \otimes G_3 \}. \end{aligned}$$

Point (3) follows by defining $H'_i = ((Y_i \leftarrow X) \otimes id_1) \circ G_i$ for $i = 1, 2$, and $H_3 = G_3$. Note that the three bigraphs G_i and H_i have open links as so does $[\![P]\!]_X$. Finally, we point (3) implies point (2), since the previous reasoning can be inverted. $\square$

By following the ideas of [22] it is easy to demonstrate that there is an exact match between reaction relations generated in CCS and in the bigraphical system, as stated in the following lemma.

Proposition 3 (Matching Reactions). *For every finite set of names X it holds*

$$P \to Q \quad \textit{if and only if} \quad [\![P]\!]_X \longrightarrow [\![Q]\!]_X$$

for every CCS process P and Q such that $Act(P), Act(Q) \subseteq X$.

Proof. For the forward direction, proceed by induction on the number of the rules applied in the derivation for $P \to Q$ in CCS. The base of the induction is the only rule without premixes, that means P is $a.P_1 \mid \overline{a}.P_2$ and Q is $P_1 \mid P_2$. The translation is sound as regards this rule, since the reactive system says

$$((\mathsf{act}_a \mid id_X) \circ [\![P_1]\!]_X) \mid ((\mathsf{coact}_a \mid id_X) \circ [\![P_2]\!]_X) \longrightarrow P$$

Table 6.2. *Semantics of formulae* L_{spat} *in CCS*

$P \models_{spat}$	0	if $P \equiv \mathbf{0}$
$P \models_{spat}$	$\neg A$	if not $P \models_{spat} A$
$P \models_{spat}$	$A \wedge B$	if $P \models_{spat} A$ and $P \models_{spat} B$
$P \models_{spat}$	$A \mid B$	if there exist R, Q, s.t. $P \equiv R \mid Q$, $R \models_{spat} A$ and $Q \models B_{spat}$
$P \models_{spat}$	$A \triangleright B$	if for every Q, $Q \models_{spat} A$ implies $P \mid Q \models_{spat} B$
$P \models_{spat}$	$\blacklozenge A$	if there exist P' s.t. $P \longrightarrow P'$ and $P' \models_{spat} A$

P_i such that $[\![P_i]\!]$ corresponds to G_i, hence $[\![P]\!] \equiv [\![a.P_1 \mid \overline{a}.P_2 \mid P_3]\!]$ and $[\![Q]\!] \equiv [\![P_1 \mid P_2 \mid P_3]\!]$. Again, Lemma 7 says that $P \equiv a.P_1 \mid \overline{a}.P_2 \mid P_3$ and $Q \equiv P_1 \mid P_2 \mid P_3$, then $R \to Q$. □

It can be proved an even stronger result: if a CCS translation reacts to a bigraph, then such a bigraph is a CCS translation as well, as formalised in the lemma below.

Proposition 4 (Conservative Reaction). *For every CCS process P such that* $[\![P]\!]_X \longrightarrow G$*, there exists a CCS process Q such that* $[\![Q]\!]_X = G$ *and* $P \to Q$.

Proof. Assume that $[\![P]\!]_X \longrightarrow G$, then the point (2) of Lemma 8 says that G has type $\epsilon \to \langle 1, X \rangle$ and open links, since so does $[\![P]\!]_X$. This means, by Lemma 7, that there exists a process Q such that $[\![Q]\!]_X \equiv G$. Conclude $P \to Q$ by Lemma 3. □

The work [2] introduces the spatial logic L_{spat} suitable to describe the structure and the behaviour of CCS processes. The language of the logic is

$$A, B \quad ::= \quad 0 \quad \mid \quad A \wedge B \quad \mid \quad A \mid B \quad \mid \quad \neg A \quad \mid \quad A \triangleright B \quad \mid \quad \blacklozenge A.$$

It includes the basic spatial operators: the void constant 0, the composition operator $|$, and its adjunct operator $\triangleright$. It presents also a temporal operator, the next step modality $\blacklozenge$, to capture the dynamics of the processes. The paper [2] defines a semantics to L_{spat} in term of CCS processes, as outlined in Tab. 6.2. In particular, the parallel connective describes processes that are produced by the parallel between two processes that satisfies the corresponding formula. A process satisfies the formula $A \triangleright B$ if it satisfied the formula B whenever put in parallel with a process satisfying A. Finally the next step $\blacklozenge A$ is satisfied by a process that can evolve into a process satisfying A.

The logic L_{spat} can be encoded in a suitable instantiation of BiLog, without using the modality defined in (3). It is sufficient to instantiate the logic BiLog$(M, \otimes, \epsilon, \Theta, \equiv, \tau)$ to obtain the bigraphical encoding of CCS. We define Θ to be composed by the standard constructor for a bigraphical system with $\mathsf{K} = \{\mathsf{act}, \mathsf{coact}\}$, and the transparency predicate τ to be always true. The fact that τ is verified on every term is determinant for the soundness of the encoding we are describing.

Rephrasing Lemma 8 informally, we say that the set of reactions in CCS are determined by couples of the form $(Redex_a, Reactum_a)$ for every $a \in X$, and every reacting process is characterised by

$$[\![P]\!]_X$$

the power of the somewhere operator. We will show that a bigraph satisfies $[\![P]\!]_X \models [\![A □ B]\!]_X$ if it satisfies $[\![B]\!]_X$ whenever connected in parallel with any encoding of a CCS process satisfying $[\![A]\!]_X$.

On the other side, in the encoding for the temporal modality ♦ the supporting formula **Triple** is satisfied by processes that are the composition of three single-rooted ground bigraphs whose outerfaces have the same number of names as X. We will show that a process satisfies $[\![\lozenge A]\!]_X$ if and only if it is the combination of a particular redex with a bigraph that satisfies the requirement of Lemma 8, and moreover that the corresponding reactum satisfies $[\![A]\!]_X$.

The main result of this section is formalised in Proposition 5. It expresses the semantical equivalence between L_{spat} and its encoding in BiLog. Note in particular the requirement for a finite set of actions performable by the CCS processes. Such a limitation is not due to the presence of the next step operator. Indeed, looking carefully at the proof, one can see that the induction step for the temporal operator still holds in the case of a not-finite set of actions. On the P = contrary, the limitation is due to the adjoint operator □. In fact we need to bound the number of names that is shared between the processes. This happens because of the different choice for the logical product operator in BiLog. On one hand, the spatial logic had the parallel operator built in. This means that the logic does P. not care about the names that are actually shared between th tthat thate the d4u9iI

$[\![A]\!]_X$, and this means $Q \models_{spat} A$ by induction hypothesis. We conclude that $[\![P]\!]_X \models [\![\blacklozenge A]\!]_X$ is equivalent to $P \to Q$ with $Q \models_{spat} A$, namely $P \models_{spat} \blacklozenge A$. □

7 Conclusions and future work

This paper moves a first step towards describing global resources by focusing on bigraphs. Our final objective is to design a general dynamic logic able to cope uniformly with all the models bigraphs have been proved useful for, as of today these include λ-calculus [21], Petri-nets [20], CCS [22], pi-calculus [16] and preserves decidability in spatial logics [11].

We have not addressed a logic for tree with hidden names. As a matter of fact, we have such a logic. More precisely we can encode abstract trees into bigraphs with an unique control amb with arity one. The name assigned to this control will actually be the name of the ambient. The extrusion properties and renaming of abstract trees have their correspondence in bigraphical terms by means of substitution and closure properties combined with properties of identity.

BiLog can express properties of trees with names. At the logical level we may encode operators of tree logic with hidden names as follows:

$$
\begin{aligned}
©\, a &\overset{def}{=} ((a \leftarrow a) \;\square\; \mathbf{id}) \circ \mathbf{T} \\
\mathbf{C}x.\, A &\overset{def}{=} \text{И}x.\, (/x \;\square\; \mathbf{id}) \circ A \\
a \;®\; A &\overset{def}{=} (\neg ©\, a \;\square\; A) \;\square\; (/a \;\square\; \mathbf{id}) \circ A \\
\mathbf{H}x.\, A &\overset{def}{=}
\end{aligned}
$$

[3] C. Calcagno, L. Cardelli, and A. D. Gordon. Deciding validity in a spatial logic for trees. In *Proc. of ACM SIGPLAN Workshop on Types in Language Design and Implementation (TLDI)*, pages 62 – 73. ACM Press, 2003.

[4] C. Calcagno, P. Gardner, and U. Zarfaty. A context logic for tree update. In *Proc. of ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 271–282. ACM Press, 2005.

[5] L. Cardelli, P. Gardner, and G. Ghelli. A spatial logic for querying graphs. In *Proc. of International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2380 of *LNCS*, pages 597 – 610. Springer-Verlag, 2002.

[22] R. Milner. Pure bigraphs. Technical Report UCAM-CL-TR-614, University of Cambridge, January 2005.

[23] Peter O'Hearn, John C. Reynolds, and Hongseok Yang. Local reasoning about programs that alter data structures. In *Proc. of International Workshop on Computer Science Logic (CSL)*, volume 2142 of *LNCS*, pages 1–19. Springer-Verlag, 2001.

[24] A. M. Pitts. Nominal logic: a first order theory of names and binding. In *Proc. of International Symposium on Theoretical Aspects of Computer Software (TACS)*, volume 2215 of *LNCS*, pages 219–242. Springer-Verlag, 2001.

[25] D. Sangiorgi. Extensionality and intensionality of the ambient logic. In *Proc. of ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 4–13. ACM Press, 2001.