On hierarchical graphs: reconciling bigraphs, gs-monoidal theories and gs-graphs*

Roberto Bruni[1], Ugo Montanari[1], Gordon Plotkin[2], and Daniele Terreni[1]

[1] Computer Science Department, University of Pisa, Italy
[2] LFCS, School of Informatics, University of Edinburgh, UK

Abstract. Compositional graph models for global computing systems must account for two relevant dimensions, namely *nesting* and *linking*. In Milner's *bigraphs* the two dimensions are made explicit and represented as loosely coupled structures: the *place graph* and the *link graph*. Here, bigraphs are compared with an earlier model, gs-graphs, based on gs-monoidal theories and originally conceived for modelling the syntactical structure of agents with α-convertible declarations. We show that gs-graphs are quite convenient also for the new purpose, since the two dimensions can be recovered by introducing two types of nodes. With respect to bigraphs, gs-graphs can be proved essentially equivalent, with minor differences at the interface level. We argue that gs-graphs have a simpler and more standard algebraic structure for representing both states and transitions, and can be equipped with a simple type system (in the style of relational separation logic) to check the well-formedness of *bounded* gs-graphs. Another advantage concerns a textual form in terms of sets of assignments, which can make implementation easier in rewriting frameworks like Maude. Vice versa, the reactive system approach developed for bigraphs needs yet to be addressed in gs-graphs.

1 Introduction

When modelling distributed systems, it is necessary to represent states and their evolutions. Usually, states are seen as terms of an algebra equipped with certain structural axioms, and state transitions are defined via conditional term rewriting rules in the SOS format. A different alternative is to represent states as graphs and transitions as graph transformations. To have the best of the two approaches, it is sometimes possible to characterise the terms up to structural axioms as graphs and the transitions derivable via the SOS inference rules as graph transformations. A good example is provided by arrows of gs-monoidal theories, which can be seen as gs-graphs, and transitions, represented as 2/double cells of a 2/double category, which can be seen as gs-graph transformations. The approach has been applied to π-calculus [9], to CCS with localities [9] and causality [2], to logic programming [3] and to other models of computation.

* Research supported by the EU Integrated Project 257414 ASCENS and by the Italian MIUR Project IPODS (PRIN 2008).

Gs-monoidal theories [6, 2] are suitable symmetric strict monoidal categories [14] that resemble cartesian (Lawvere) theories, but without the two naturality axioms that allow copying shared subterms and garbage collecting unused terms. In addition, gs-monoidal theories are built out of symbols taken from a hyper-signature instead of an ordinary signature. The difference is that symbols in a hyper-signature are not constrained to have single-sorted codomain, but, like domains, can be a tuple of sorts. The corresponding gs-graphs are a kind of dags where substructures can be shared due to both ordinary duplicators, as in term graphs, and hyper-signature symbols (see e.g. Fig. 2(a)). A useful feature of gs-graphs is that they can be represented in textual form as sets of assignments of the form $x, y := f(z, v)$, where f is a signature symbol that takes two arguments and returns a pair of results and x, y, z, v are α-convertible names. Here 2-cells are essentially like term rewriting rules which can be both contextualised and instantiated. Double cells allow one to model synchronisation of local rewritings.

Recent developments in the area of open-ended systems for global computing have emphasised the need for *hierarchical* graph models: they have two relevant dimensions, namely *nesting* and *linking*. The former has to do with the structural design of processes (e.g., the scoping of a transaction, a compensation, or a session, or the containment of an ambient, a membrane, or an environment); it induces a tree-like hierarchy on nodes. The latter concerns interaction capabilities (e.g., for communication, handshaking, or connectivity) that are flat, and may connect any tree nodes. This is the *pure* case. The situation is more complex in the *binding* case, where a name used for communication is declared at some level in the tree hierarchy and is usable only below its declaration point. Inspired by Cardelli and Gordon's *ambients* [4] and aiming at defining a general, flexible and easy to grasp model, Milner defined *bigraphs*, where the two dimensions are made explicit and represented as essentially orthogonal structures, called *place graph* and *link graph*. Bigraphs [19, 18, 13, 16] have been studied in depth from several points of view, in particular as a basis of the *reactive system* approach, where a labelled transition system, over which bisimilarity is a congruence, is synthesised from a reduction semantics. This part of the theory is not covered for gs-graphs.

Gs-graphs were originally conceived with the syntactical structure of agents with α-convertible declarations in mind, not the hierarchical structure of global computing systems. However it turns out that they are also quite convenient for this new purpose. The *nesting* and *linking* structure can be recovered by defining two types of nodes, *black* nodes, which represent intermediate places in the tree-like hierarchy, and *white* nodes which are communication names/channels.

In this paper, gs-graphs are compared with support equivalent bigraphs. Our first correspondence result states that the two models essentially coincide in the pure case. The only difference is in the interface: gs-graphs have an ordered tuple of connectors and all the names are α-convertible, while bigraph connectors are decorated with names. The "names versus strings in a free monoid" dichotomy can already be found in the simple case of equational theories versus Lawvere

theories, where one has to translate between variables and numbers. On the one hand, named connectors facilitate the direct representation in bigraphs of standard process calculi operations; on the other hand, they make the algebraic structure of bigraphs more complex and less standard: e.g. parallel composition is partial and sequential composition is associative only up to isomorphism. Also the rewriting structure does not generate composable cells. On the contrary, gs-graphs inherit from gs-monoidal theories a variety of well-behaved operations.

Our second correspondence result is concerned with (support equivalent) binding bigraphs. In the binding case, it is necessary to constrain the compositional structure to generate only *legal* graphs. For bigraphs [8], additional information is inserted in the interface to allow for a complete axiomatisation. In our approach, we introduce a type system which recognises legal binding gs-graphs. On gs-monoidal theories the type system is represented by membership sentences in membership equational logic [15], while on gs-graphs we exploit a quite simple relational type system (in the style of *relational separation logic* [21]). When the gs-graph is pure, no pairs are generated; parallel composition does not add any pair; and for sequential composition existing pairs are preserved, but new pairs, possibly leading to inconsistency, are generated. The complexity of the proposed typing algorithm is $O(B \cdot W)$, where B is the number of black nodes and W is the number of bound white nodes.

The formal assessment of the analogies and differences between the two different proposals and the definition of transformations to move from one framework to the other allows us to conclude that: (i) bigraphs can be presented at a suitable level of abstraction as arrows of a particular free symmetric monoidal theories, in a perfectly standard way; and (ii) the gs-graphs representation seems to offer some advantages over the others.

Structure of the paper. Section 2 recaps the basics of the models we are comparing. Section 3 addresses the case of pure signatures, while the binding case is discussed in Section 4. Finally, Section 5 contains come concluding remarks.

Additional material is concerned with the formal definition of sequential and parallel compositions of gs-graphs (Appendix A) and of bigraphs (Appendix B), their preservation via the transformations presented in the paper (Appendix C) and the technical details of the transformation from binding bigraphs to gs-graphs (Appendix D).

2 Background on graph-based structures

Notation. For an ordinal n, we write $i \in n$ as a shorthand for $i \in \{0, \ldots, n-1\}$ and let $[n, m]$ denote the set $\{i \mid n \leq i \leq m\}$. We use the symbol $\uplus$ for disjoint union of sets. We let S^* denote the free monoid over the elements in S, whose product is juxtaposition and whose unit is denoted by ϵ. We abbreviate the juxtaposition of n consecutive objects u by u^n, with $u^0 = \epsilon$. We overload $|\,\boldsymbol{.}\,|$ to denote the length of a string, the cardinality of a set and the *support* of place / link / bigraphs (see Definitions 6–8).

$$\text{(ops)}\ \frac{f \in \Sigma_{u,v}}{f : u \to v} \qquad \text{(ids)}\ \frac{u \in S^*}{id_u : u \to u} \qquad \text{(bang)}\ \frac{u \in S^*}{!_u : u \to \epsilon} \qquad \text{(dup)}\ \frac{u \in S^*}{\nabla_u : u \to uu}$$

$$\text{(sym)}\ \frac{u, v \in S^*}{\rho_{u,v} : uv \to vu} \qquad \text{(seq)}\ \frac{t : u \to v \quad t' : v \to w}{t; t' : u \to w} \qquad \text{(par)}\ \frac{t : u \to v \quad t' : u' \to v'}{t \otimes t' : uu' \to vv'}$$

Fig. 1. Inference rules of gs-monoidal theories

2.1 From signatures to gs-graphs

The gs-monoidal approach is based on representing basic computational entities and resources as hyperedges and interaction capabilities by the way in which their tentacles are connected to nodes. (The name *gs* comes after *graph structure*.) For example, nodes can model communication channels and tentacles can express the capability to perform i/o operations on them.

The idea is to consider a particular class of graphs obtained by selecting a few basic shapes for hyper-edges (i.e. fix a hyper-signature) and by freely composing them in series and in parallel to build larger and more complex shapes. Moreover, it is allowed: 1) to rearrange the wirings of tentacles to connect in series edges that otherwise are not "adjacent"; 2) to mark nodes as private to a certain subgraph so that no other tentacle can be attached to them; 3) to attach more than two tentacles to the same node. As only acyclic structures are allowed, hyperedges can be stratified along the implicit partial order defined by tentacle connections.

Definition 1 (hyper-signature). *Given a set S of sorts, a* hyper-signature *(*signature*, for short) Σ is a family $\{\Sigma_{u,v}\}_{u,v \in S^*}$ of sets of operators such that each $f \in \Sigma_{u,v}$ takes $|u|$ arguments typed according to u and returns a tuple of $|v|$ values typed according to v.*

The expressions of interest are generated by the rules depicted in Fig. 1. By rule (ops), the basic expressions include one generator for each operator of the signature. All other basic terms define the wires that can be used to build our graphs: the *identities* (ids), the *dischargers* (bang), the *duplicators* (dup) and the *symmetries* (sym). These are the elementary bricks of our expressions, and we get the remaining ones by closing them with respect to *sequential* (seq) and *parallel* (par) *composition*. Every expression $t\colon u \to v$ generated by the inference rules is typed by a *source* and by a *target* sequence of sorts (u and v, respectively), which are relevant only for the sequential composition, which is a partial operation. A *wiring* is an arrow of $\mathbf{GS}(\Sigma)$ which is obtained from the rules of Fig. 1 without using rule (ops).

Definition 2 (gs-monoidal theory). *Given a signature Σ over a set of sorts S, the* gs-monoidal theory $\mathbf{GS}(\Sigma)$ *is the (symmetric, strict monoidal) category whose objects are the elements of S^* and whose arrows are equivalence classes of* gs-monoidal terms*, i.e., terms generated by the inference rules in Fig. 1 subject to the following conditions*

- *identities and sequential composition satisfy the axioms of categories*
  [identity] $id_u \,;\, t = t = t \,;\, id_v$ *for all* $t : u \to v$*;*
  [associativity] $t_1 \,;\, (t_2 \,;\, t_3) = (t_1 \,;\, t_2) \,;\, t_3$ *whenever any side is defined,*
- $\otimes$ *is a monoidal functor with unit* id_ϵ*, i.e., it satisfies*
  [monoid] $t \otimes id_\epsilon = t = id_\epsilon \otimes t \quad t_1 \otimes (t_2 \otimes t_3) = (t_1 \otimes t_2) \otimes t_3$
  [functoriality] $id_{uv} = id_u \otimes id_v$*, and*
  $(t_1 \otimes t_2) \,;\, (t_1' \otimes t_2') = (t_1 \,;\, t_1') \otimes (t_2 \,;\, t_2')$ *whenever both sides are defined,*
- ρ *is a symmetric monoidal natural transformation, i.e., it satisfies*
  [naturality] $(t \otimes t') \,;\, \rho_{v,v'} = \rho_{u,u'} \,;\, (t' \otimes t)$ *for all* $t : u \to v$*,* $t' : u' \to v'$
  [symmetry]
  $(id_u \otimes \rho_{v,w}) \,;\, (\rho_{u,w} \otimes id_v) = \rho_{uv,w} \quad \rho_{\epsilon,u} = \rho_{u,\epsilon} = id_u \quad \rho_{u,v} \,;\, \rho_{v,u} = id_{uv}$
- ∇ *and* $!$ *satisfy the following axioms*
  [monoidality] $\nabla_{uv} \,;\, (id_u \otimes \rho_{v,u} \otimes id_v) = \nabla_u \otimes \nabla_v \quad !_{uv} = !_u \otimes !_v$
  [unit and duplication] $!_\epsilon = \nabla_\epsilon = id_\epsilon \quad \nabla_u \,;\, \rho_{u,u} = \nabla_u$
  $\nabla_u \,;\, (id_u \otimes \nabla_u) = \nabla_u \,;\, (\nabla_u \otimes id_u) \quad \nabla_u \,;\, (id_u \otimes !_u) = id_u$

Remark 1. We let $\otimes$ take precedence over ;. We shall focus on two-sorted signatures over $S = \{\bullet, \circ\}$, where $\bullet$ nodes are used for locations, while $\circ$ nodes for links. Furthermore, for ease of modelling bigraphs, we reverse the sense of direction for composing arrows, i.e. we take cogs-monoidal theories. As a matter of notation we swap implicitly the source and target of each arrow, e.g. letting

$$\rho_{\bullet,\circ} : \circ\bullet \to \bullet\circ \qquad \nabla_\bullet : \bullet^2 \to \bullet \qquad !_\circ : \epsilon \to \circ.$$

Moreover, we assume all signatures include the operator $\nu : \circ \to \epsilon$. Note that the expression $!_\circ \,;\, \nu : \epsilon \to \epsilon$ denotes a special arrow that is the counterpart of so-called *idle edges* in bigraphs jargon. While the axiom $!_\circ \,;\, \nu = id_\epsilon$ can be useful in many situations, we decide not to impose it here, because it is not standard for gs-monoidal theories. This point is further discussed in Section 5.

Example 1. Let us consider a (cogs) signature with three operators $f, h : \bullet \to \bullet\circ$ and $g : \bullet \to \bullet\circ^2$. Then we can compose, e.g., the expressions below:

$$\begin{aligned}
e_1 &\triangleq f \otimes id_\circ \,;\, id_\bullet \otimes \nabla_\circ : \bullet\circ \to \bullet\circ \\
e_2 &\triangleq (!_\bullet \,;\, h) \otimes (!_\bullet \,;\, h) \,;\, id_\bullet \otimes \rho_{\bullet,\circ} \otimes id_\circ \,;\, \nabla_\bullet \otimes \nabla_\circ : \epsilon \to \bullet\circ \\
e_3 &\triangleq g \otimes id_\circ \,;\, \rho_{\circ,\bullet} \otimes \rho_{\circ,\circ} : \bullet\circ \to \circ\bullet\circ^2 \\
e_4 &\triangleq e_1 \otimes (e_2 \,;\, e_3) \,;\, id_\bullet \otimes (\nabla_\circ \,;\, \nu) \otimes id_{\bullet\circ\circ} : \bullet\circ \to \bullet^2\circ^2
\end{aligned}$$

We note that a cogs-monoidal theory is a *sm Lawvere theory* [11] in which every sort is a commutative monoid.

The algebraic structure of gs-monoidal theories finds suitable realisation in graph-based modelling: arrows can be interpreted as *concrete* acyclic directed hypergraphs with interfaces, taken up to renaming of their nodes; all such graphs are represented by some arrow; any two isomorphic graphs whose interfaces match are represented by the same arrow and are thus equivalent *abstract* graphs.

We find it convenient to represent concrete gs-graphs as sets of *assignments*. We assume V is a denumerable set of *S-sorted names*, equipped with a total

order $\leq$ and such that there are infinitely many names for each sort. Names are denoted by $n_1 : s_1, n_2 : s_2, \ldots$ or simply by $n_1, n_2, \ldots$ when the sort is clear from the context. A *name substitution* is a sort-preserving morphism $\sigma : V \to V$.

Remark 2. When the sort $S = \{\bullet, \circ\}$ is considered, we use $p, q, ...$ for names of sort $\bullet$ and $x, y, z, ...$ for names of sort $\circ$. When needed, we assume the order $\leq$ is induced by subscripts, i.e., that $n_i \leq n'_j$ iff $i \leq j$.

Definition 3 (assignment, multi-assignment). *Let $n'_i : s'_i$ for $i \in [1, k]$, $n_j : s_j$ for $j \in [1, h]$, $u = s'_1 \ldots s'_k$ and $v = s_1 \ldots s_h$. A* proper assignment *is written $n'_1 \ldots n'_k := f(n_1, \ldots, n_h)$ where $f \in \Sigma_{u,v}$, When $f \in \Sigma_{u,\epsilon}$ the assignment is written as $n'_1, \ldots, n'_k := f$, while when $f \in \Sigma_{\epsilon,v}$ it is written $f(n_1, \ldots, n_h)$. An* auxiliary assignment *is written either $n := n'$ (aliasing), with n and n' having the same sort, or $!(n)$ (name disposal). A* multi-assignment *G is a multiset of (proper and auxiliary) assignments.*

When a name appears in the left member of an assignment we say that it is *assigned*, when it appears in the right member we say that it is *used*. For an auxiliary assignment $n := n'$ we say n is an *inner connection* of the interface. The set of *outer connections* of a multi-assignment consists of all names that are used but not assigned. We denote with $ic(G)$ (resp. $oc(G)$) the list of the inner (resp. outer) connections of a multi-assignment G (ordered according to $\leq$). We say that $n \sqsubset_G n'$ if G contains an assignment where n is used and n' is assigned.

Proper assignments define the hyperedges of the graphs, whose tentacles are attached to nodes named according to their assigned and used names. Node sharing is realised by using the same name more than once. Auxiliary assignments allow to expose more references to the same node in the interface or to prevent certain nodes from appearing in the interface.

Definition 4 (gs-graph). *A* concrete gs-graph *is a multi-assignment G s.t.: (1) every name is assigned at most once; (2) the transitive closure $\sqsubset_G^+$ of $\sqsubset_G$ is irreflexive; (3) every $n \in ic(G)$ is a maximal element of $\sqsubset_G^+$; (4) for each name $n \notin ic(G)$ (exactly) one assignment $!(n)$ is present. Two concrete gs-graphs G and H are isomorphic if H can be obtained from G by applying an injective name substitution that respects the total ordering $\leq$ of the inner and outer connections. An* abstract gs-graph *(or simply* gs-graph*) is the equivalence class of a concrete gs-graph modulo isomorphism.*

Since gs-graphs are taken up to isomorphism, the exact choice of names is immaterial. The constraints on gs-graphs allow us to introduce more concise representation of gs-graphs by: (i) an auxiliary assignment of the form $!(n)$ is omitted whenever n is used in some other assignment; (ii) an auxiliary assignment $n := n'$ is omitted if n' is not an outer connection and it is a maximal element, except for n, of the partial order $\sqsubseteq^+$;

It can be shown that the gs-graphs defined over a signature Σ form a (symmetric monoidal) category that is (naturally) isomorphic to the gs-monoidal theory of Σ (see [9]). The idea is that a gs-graph G whose lists of sorts of inner

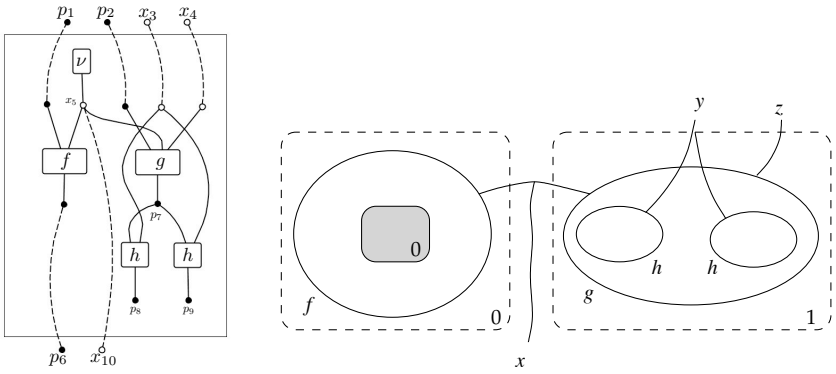

Fig. 2. Different graphical models for the same structure

connections, $ic(G)$, and outer connection, $oc(G)$ are u and v, respectively, can be regarded as an arrow $G : u \to v$. Then we can fix *atomic* gs-graphs for the basic building blocks of gs-monoidal theories and define how to compose gs-graphs in sequence $G_1; G_2$ and in parallel $G_1 \otimes G_2$ (see Appendix A).

Example 2. The arrow e_4 from Example 1 corresponds to the gs-graph $G = \{\ x_5 := \nu\ ,\ p_6 := f(p_1, x_5)\ ,\ p_7 := g(p_2, x_5, x_4)\ ,\ p_8 := h(p_7, x_3)\ ,\ p_9 := h(p_7, x_3)\ ,\ x_{10} := x_5\ ,\ !(p_8)\ ,\ !(p_9)\ \}$.

2.2 From signatures to bigraphs

The separation between different concerns is made more explicit in *bigraphs*, which are composed by two graphs, the *place graph* and the *link graph*, defined on the same set of nodes. In the literature two main classes of bigraphs have been developed: the *pure bigraphs* [12] and the *binding bigraphs* [17].

In pure bigraphs a node is not allowed to declare a local name, and the nodes communicate using only their global ports.

Definition 5 (pure signature). *A* pure signature *consists of a set $\mathcal{K}$ whose elements, called* controls*, specify the role of system nodes and a function $ar : \mathcal{K} \to \mathbb{N}$ that assigns an* arity *to each control, i.e. the number of its ports.*

A place graph is essentially a forest of unordered trees, and represents the *locality* of the nodes, that is where they are placed in the hierarchy.

Definition 6 (place graph). *Let $\mathcal{K}$ be a pure signature and m, n be a pair of ordinals, then a place graph $P : m \to n$ is a triple $(V_P, ctrl_P, prnt_P)$ where V_P is a finite set of nodes called the* support *of P (also denoted $|P|$), $ctrl_P : V_P \to \mathcal{K}$ is the* control map *assigning a control to each node and $prnt_P : m \uplus V_P \to V_P \uplus n$ is the* parent map *that describes the location of the nodes. The parent map is acyclic in the sense that for each $v \in V$ $prnt^k(v) = v$ implies $k = 0$.*

A link graph is a graph expressing the *connectivity*: an edge represent e.g. a communication medium between attached nodes.

Definition 7 (link graph). *Given a pure signature $\mathcal{K}$ a* link graph $L = (V_L, E_L, ctrl_L, link_L) : X \to Y$*, where X and Y are the sets respectively of* inner *and* outer names*, has finite sets of nodes V_L and edges E_L, a control map $ctrl_L : V_L \to \mathcal{K}$ and a* link map *$link : X \uplus P_L \to E_L \uplus Y$ with $P_L \triangleq \sum_{v \in V_L} ar(ctrl(v))$ the set of ports of L. The* support *of L is $|L| \triangleq V_L \uplus E_L$.*

The key point of bigraphs is that their place and link graphs are constructed separately; therefore the locality of a node and its connectivity can not interfere.

Definition 8 (concrete pure bigraph). *A* concrete (pure) bigraph $G = (V_G, E_G, ctrl_G, prnt_G, link_G) : \langle m, X\rangle \to \langle n, Y\rangle$ *consists of a place graph $G^P = (V_G, ctrl_G, prnt_G) : m \to n$ and a link graph $G^L = (V_G, E_G, ctrl_G, link_G) : X \to Y$. It is* lean *if it has no idle edges. We sometimes write $G = \langle G^P, G^L\rangle$ and the* support *of G is $|G| \triangleq V_G \uplus E_G$.*

Example 3. An example of pure bigraph is in Fig. 2(b). The place graph is represented through the nesting of nodes, while the arcs pertain to the link graph. The interface is given by pairing the interfaces of the place graph and of the link graph. The outer interface of a place graph specifies the number of distinct components forming the bigraph; to each component corresponds a *root* displayed with an enclosing dashed box. In the example we have two roots (numbered 0 and 1). The inner interface consists of the holes of the place graph, called *sites*, that serve to compose with other place graphs. Our example has one hole (numbered 0), displayed with a grey box. For the link graph, outer names are displayed on the top (y and z), and inner names on the bottom (x).

Definition 9 (support equivalence for bigraphs). *Given two bigraphs $G, H : \langle m, X\rangle \to \langle n, Y\rangle$, a support equivalence $\rho : |G| \to |H|$ is a pair of bijections $\rho_V : V_G \to V_H$ and $\rho_E : E_G \to E_H$ such that: $ctrl_H \circ \rho_V = ctrl_G$, $prnt_H \circ (Id_m \uplus \rho_V) = (Id_n \uplus \rho_V) \circ prnt_G$ and $link_H \circ (Id_X \uplus \rho_P) = (Id_Y \uplus \rho_E) \circ link_G$, where $\rho_P : P_G \to P_H$ maps the ports of G in those of H and it is defined in terms of ρ_V: for each port $(v, i) \in P_G$, $\rho_P((v, i)) = (\rho_V(v), i)$.*

We write $G \simeq H$ when G and H are *support equivalent*, and $G \eqslantless H$ if they are support equivalent ignoring their idle edges (*lean-support equivalence*). The lean-support quotient yields the (partial) symmetric monoidal category of *abstract bigraphs*, denoted by $BG(\mathcal{K})$, and the lean-support quotient functor $[\![\cdot]\!]$ maps each concrete bigraph in its corresponding abstract bigraph.

Definition 10 (abstract bigraphs). *An* abstract bigraph *over $\mathcal{K}$ is a $\eqslantless$-equivalence class $[G] : \langle m, X\rangle \to \langle n, Y\rangle$ of a concrete bigraph G.*

In binding bigraphs, besides the possibility of having local names, there is added a scope discipline for limiting the visibility of such local names. In particular a control may declare some names that only its descendants can use, thus relaxing in part the assumption of independence of the two graphical structures.

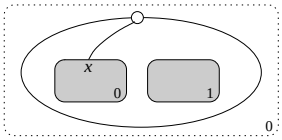

Fig. 3. A binding bigraph

Definition 11 (binding signature). *A* binding signature *has a set of controls* $\mathcal{K}$ *and an* arity *function* $ar : \mathcal{K} \to \mathbb{N} \times \mathbb{N}$. *If* $ar(K) = (h, k)$, *we write* $K : h \to k$ *and we call, respectively,* h *and* k *the* binding arity *and the* free arity *of* K *and they index respectively the* binding ports *and the* free ports *of* K.

Definition 12 (binding interface). *A* binding interface *is a triple* $I = \langle m, loc, X\rangle$ *where the ordinal* m *and the set of names* X *are as in pure bigraphs, and* $loc \subseteq m \times X$ *is the* locality *of the interface. If* $(i, x) \in loc$ *we say that* i *is a* place *of* x. *We denote by* $I^u = \langle m, X\rangle$ *the pure interface underlying* I.

Example 4. The approach of the binding bigraphs for avoiding misleading compositions consists in enriching the interfaces with a locality relation *loc* that establishes to which places, if any, a name belongs to. Fig. 3 denotes a simple binding bigraph with a single control with a local name and two sites in it; the locality relation on the inner interface associates the name to the first site. This restriction prevents controls in the second site from using this name.

Definition 13 (locality relation). *Let* $I = \langle m, loc_I, X\rangle$ *and* $J = \langle n, loc_J, Y\rangle$ *be binding interfaces and consider a pure bigraph* $G^u : I^u \to J^u$ *on the pure underlying interfaces. Then the* locality relation $loc_G \subseteq (m \uplus n \uplus V) \times (X \uplus Y \uplus P \uplus E)$, *is the smallest relation such that:*

- *if* $(i, x) \in loc_I$ *then* $(i, x) \in loc_G$ *(*$loc_I \subseteq loc_G$*)*
- *if* $(j, x) \in loc_J$ *then* $(j, x) \in loc_G$ *(*$loc_J \subseteq loc_G$*)*
- *if* p *is a binding port of a node* v *then* $(v, p) \in loc_G$
- *if* p *is a free port of a node* v *then* $(prnt(v), p) \in loc_G$
- *if an edge* e *contains a binding port of* v *then* $(v, e) \in loc_G$

Definition 14 (binding bigraph). *Given two binding interfaces* I *and* J *a* concrete binding bigraph $G : I \to J$ *consists of an underlying pure bigraph* $G^u : I^u \to J^u$ *such that: (a) any edge has at most one binding port, while an outer name has none; (b) if* $link_G(q) = l$ *is a local link then* q *is also local, and whenever* $(w, q) \in loc_G$ *then there exists* w' *such that* $prnt^k_G(w) = w'$ *for some* $k \in \mathbb{N}$ *and* $(w', l) \in loc_G$. *The condition (b) is called the* scoping rule.

3 Characterising Pure Bigraphs

This section shows that pure bigraphs are *essentially* equivalent to a particular class of gs-graphs over the sorts $\{\bullet, \circ\}$. The word "essentially" means that there

is a one-to-one relation between the objects of the two models, but only up to certain bijections over the interfaces. Indeed the main difference lies in the way through which the two models view the interfaces: for a bigraph an interface is a pair composed by an ordinal and by a set of names, in a gs-graph instead the interfaces are strings over the alphabet $\{\bullet, \circ\}$. For making them comparable we need to equip each model with some missing information which is present in the other model. In a bigraphical interface $\langle m, X\rangle$ we must form a list out of the elements in $\{0, \ldots, m-1\}$ and the elements in X. For the gs-graphs instead, given a string in $\{\bullet, \circ\}^*$ we have to assign a name to each element of sort $\circ$.

The relation between pure bigraphs and gs-graphs can be sketched by looking at Fig. 2. Places correspond to hyperedges and their hierarchy is built in the gs-graph by exploiting the nodes of sort $\bullet$. Connectivity is represented by the sharing of nodes of sort $\circ$. Closed links are the $\circ$ nodes below a restriction ν. The dashed lines express which nodes are exported to the interfaces.

Interfaces. Given a bigraphical interface $\langle m, X\rangle$, every $i \in m$ is of sort $\bullet$ while the names in X are of sort $\circ$. Nevertheless if we had a gs-graph interface with exactly m elements of sort $\bullet$ and $|X|$ elements of sort $\circ$ we would not have an obvious way to map such interface in $\langle m, X\rangle$, because the $\circ$ elements are ordered unlike the names in X. Indeed, take for example the interfaces of our running example $u = \bullet^2\circ^2$ and $\langle 2, \{y, z\}\rangle$; there are two possible bijections from $\circ^2$ to $\{y, z\}$ but only one allows to establish a correct correspondence (y must match the first $\circ$ and z the second $\circ$). On the contrary, if we knew that $y < z$, the natural way would be that of choosing the right bijection. Thus we introduce a total order on the names used in the bigraphical interfaces.

Definition 15. *Let $\mathscr{X}$ be a denumerable infinite totally ordered (by $\leq$) set of names. Given a pure signature $\mathcal{K}$, we take bigraphs in which the sets of names on the interfaces are replaced by lists of names ordered through $\leq$. Given a list L we denote by $L[j]$ the $(j+1)^{th}$ element of the list for each admissible j.*

The assumption of having total ordered names makes the two type interfaces more similar, but it is not sufficient for establishing a bijective relation between them. Consider the previous example and suppose that in $\mathscr{X}$ we have $y < x < z$. The interface I can be associated, through the unique monotone bijection, to the bigraphical interface with the set $\{y, z\}$, but nothing prevents one using $\{x, z\}$ or $\{x, y\}$ instead. These considerations lead us to the following definition that embeds a particular set of names in a gs-graph interface.

Definition 16 (name choice). *Let $u \in \{\bullet, \circ\}^*$ and let $\#_u$ be the number of elements of sort $\circ$ in u. Then a* name choice *for u is an injective monotone map $\sigma_u : \#_u \to \mathscr{X}$. A gs-graph $G : u \to v$ can be equipped with two name choices σ_u, σ_v for the inner and the outer interfaces, written $G : (u, \sigma_u) \to (v, \sigma_v)$.*

Signatures. Both bigraphs and gs-graphs are based on a signature that describes the allowed operators. Therefore it is necessary to correlate the two typologies of signature. (In the rest of this section we understand that only pure signatures are considered and we use the symbol $\mathcal{K}$ also for gs-graph signatures.)

Definition 17 (equating signatures). *Consider a pure signature $\mathcal{K}$; the equivalent gs-graph signature Σ has an operator $K : \bullet \to \bullet\circ^h$ if and only if the control K of arity h is in $\mathcal{K}$.*

Graphs. With the name choices, to a string over $\{\bullet, \circ\}$ corresponds exactly one bigraphical interface, but the converse is false. Indeed a bigraphical interface over ordered names can be viewed as a concatenation of two ordered lists, the first with elements of sort $\bullet$ and the second with $\circ$-elements, while in a gs-graph interface such elements are mixed. Thus given a bigraph we can add two bijections that "shuffle" these elements as stated below:

Definition 18 (shuffled bigraphs). *A* shuffled bigraph $\langle G : \langle m, X, \phi_{in}\rangle \to \langle n, Y, \phi_{out}\rangle$ *consists of a bigraph $G : \langle m, X\rangle \to \langle n, Y\rangle$ and two bijections, $\phi_{in} : m + |X| \to m + |X|$ and $\phi_{out} : n + |Y| \to n + |Y|$, called shuffle functions, that preserve the relative order of the elements with the same sort, i.e.:*

- $\forall i, j \in m+|X|$ *if* $0 \le i \le j < m$ *or* $m \le i \le j < m+|X|$ *then* $\phi_{in}(i) \le \phi_{in}(j)$
- $\forall i, j \in n+|Y|$ *if* $0 \le i \le j < n$ *or* $n \le i \le j < n+|Y|$ *then* $\phi_{out}(i) \le \phi_{out}(j)$.

From shuffled bigraphs to gs-graphs. The first transformation that we define takes a shuffled bigraph $G = \langle V_G, E_G, ctrl_G, prnt_G, link_G\rangle : \langle m, X, \phi_{in}\rangle \to \langle l, Y, \phi_{out}\rangle$ and returns a gs-graph $H = S[\![G]\!]$ that represents it. The underlying idea is relatively simple: there is a proper assignment for each node and edge of the bigraph. In detail the edges cause the creation of assignments with the ν operator, while the nodes give assignments that describe the position of a control in the system and the interactions with the other controls, deriving it from the parent and the link map of the bigraph. In the following we denote with $\mathcal{N}_H$ the set of all names appearing in H, which is partitioned in $\mathcal{N}_H^\bullet$ and $\mathcal{N}_H^\circ$, respectively the set of all names of sort $\bullet$ and of sort $\circ$, appearing in H. Let $\mathcal{N}_H^\bullet = V_G \uplus \{s_0, \ldots, s_{m-1}\} \uplus \{r_0, \ldots, r_{l-1}\}$ and $\mathcal{N}_H^\circ = E_G \uplus \{x_0, \ldots, x_{|X|-1}\} \uplus \{y_0, \ldots, y_{|Y|-1}\}$. Note that x_i and y_j are not the names in sets X and Y, but new names used only in the gs-graph. First we need two auxiliary functions $\overline{prnt} : m \uplus V \to \mathcal{N}_H^\bullet$ and $\overline{link} : P_G \uplus X \to \mathcal{N}_H^\circ$ that translate the results of $prnt_G$ and $link_G$ into the names of the gs-graph:

$$\overline{prnt}(v) \triangleq \begin{cases} w \text{ if } prnt_G(v) = w \in V_G \\ r_i \text{ if } prnt_G(v) = i \in l \end{cases} \qquad \overline{link}(p) \triangleq \begin{cases} e \text{ if } link_G(p) = e \in E_G \\ y_i \text{ if } link_G(p) = Y[i] \end{cases}$$

Next we define the assignments of H: (1) $\forall v \in V_G$ we let $v := f(\overline{prnt}(v), \overline{link}(v, 0), \ldots, \overline{link}(v, h-1))$ where $f = ctrl_G(v)$ and h is the arity of f; (2) $\forall e \in E_G$ we let $e := \nu$; (3) $\forall i \in m$ we let $s_i := \overline{prnt}(i)$; (4) $\forall i \in |X|$ we let $x_i := \overline{link}(X[i])$ Note that $\{s_0, \ldots, s_{m-1}\} \cup \{x_0, \ldots, x_{|X|-1}\}$ are the inner connections of H while $\{r_0, \ldots, r_{l-1}\} \cup \{y_0, \ldots, y_{|Y|-1}\}$ are its outer connections. The order of these names can be retrieved using the shuffle functions: Define $\overline{\phi}_{in}^{-1} : m + |X| \to \mathcal{N}_H$ and $\overline{\phi}_{out}^{-1} : l + |Y| \to \mathcal{N}_H$ as

$$\overline{\phi}_{in}^{-1}(j) \triangleq \begin{cases} s_i & \text{if } \phi_{in}^{-1}(j) = i < m \\ x_{i-m} & \text{if } \phi_{in}^{-1}(j) = i \ge m \end{cases} \qquad \overline{\phi}_{out}^{-1}(j) \triangleq \begin{cases} r_i & \text{if } \phi_{out}^{-1}(j) = i < l \\ y_{i-l} & \text{if } \phi_{out}^{-1}(j) = i \ge l \end{cases}$$

Hence $ic(H) = (\overline{\phi}_{in}^{-1}(0), \overline{\phi}_{in}^{-1}(1), \ldots, \overline{\phi}_{in}^{-1}(m + |X| - 1))$ and $oc(H) = (\overline{\phi}_{out}^{-1}(0), \overline{\phi}_{out}^{-1}(1), \ldots, \overline{\phi}_{out}^{-1}(l + |Y| - 1))$. Finally, the transformation $S[\![G]\!]$ produces two name choices: $\sigma_{in}(i) \triangleq X[i]$ for $i \in |X|$ and $\sigma_{out}(i) \triangleq Y[i]$ for $i \in |Y|$.

From gs-graphs to shuffled bigraphs. Let $H : (u, \sigma_u) \to (v, \sigma_v)$ be a gs-graph over $\mathcal{K}$ with name choices and let us take an instance in its isomorphism class. The first step in transforming the gs-graph H in the corresponding shuffled bigraph $G = B[\![H]\!]$ consists in defining the shuffle functions ϕ_{in} and ϕ_{out}. For this purpose let m and l be the number of elements of sort $\bullet$ in the lists u and v respectively; then for each list, for example u, define a function $u^\bullet : m \to |u|$ that tell us the positions in the list u of the $\bullet$ sort elements, and a similar function $u^\circ : (|u| - m) \to |u|$ that do the same thing on the elements of u of sort $\circ$. For example if $u = \bullet \circ \circ \bullet$, then $u^\bullet = \{0 \mapsto 0, 1 \mapsto 3\}$ and $u^\circ = \{0 \mapsto 1, 1 \mapsto 2\}$. With the aid of this functions we define $\phi_{in} : |u| \to |u|$ and $\phi_{out} : |v| \to |v|$ as:

$$\phi_{in}(i) \triangleq \begin{cases} u^\bullet(i) & \text{if } 0 \le i < m \\ u^\circ(i - m) & \text{otherwise} \end{cases} \qquad \phi_{out}(i) \triangleq \begin{cases} v^\bullet(i) & \text{if } 0 \le i < l \\ v^\circ(i - l) & \text{otherwise} \end{cases}$$

Now recall that in a pure signature for gs-graphs every operator, except ν, is of the form $f : \bullet \to \bullet\circ^h$ for some $h \in \mathbb{N}$, thus every proper assignment over those operators takes the form $n := f(n_\bullet, n_0, \ldots, n_{h-1})$ with $n, n_\bullet$ of sort $\bullet$ and the remaining names of sort $\circ$. Then, the bigraph associated to the gs-graph $H : (u, \sigma_u) \to (v, \sigma_v)$ is $G = B[\![H]\!] = (V_G, E_G, ctrl_G, prnt_G, link_G) : \langle m, X, \phi_{in}\rangle \to \langle l, Y, \phi_{out}\rangle$ where $m, l, \phi_{in}, \phi_{out}$ are defined as above, and:

- $X[i] \triangleq \sigma_u(i)$ for each admissible i and $Y[j] \triangleq \sigma_v(j)$ for each admissible j.
- $V_G \triangleq \{n \in \mathcal{N}_H^\bullet \mid n \notin ic(H) \text{ and } n \notin oc(H)\}$ is composed by the $\bullet$-names that are not visible outside the gs-graph. Thus the names in V_G are assigned exactly once with a proper assignment.
- $E_G \triangleq \{n \in \mathcal{N}_H^\circ \mid n := \nu \in H\}$ comprises all "restricted" names of sort $\circ$.
- The control map $ctrl_G : V_G \to \mathcal{K}$ is defined as follows. Being $n \in V_G$ let $n := f(n_\bullet, n_0, \ldots, n_{h-1})$ the unique assignment of n in H, then $ctrl_G(n) = f$
- The parent map $prnt_G : m \uplus V_G \to V_G \uplus l$ is defined separately for the elements in V_G and m. For each $n \in V_G$ let $n := f(n_\bullet, n_0, \ldots, n_{h-1})$ the unique assignment of n in H, then:

  $$prnt_G(n) = \begin{cases} n_\bullet & \text{if } n_\bullet \in V_G \\ \phi_{out}^{-1}(j) & \text{if } n_\bullet = oc(H)[j] \text{ for some } j \text{ in the list range} \end{cases}$$

  Take instead $i \in m$ and let $s_i = ic[\phi(i)]$. Since s_i is an inner connection, there exists in H a unique auxiliary assignment $s_i := n$.

  $$prnt_G(i) = \begin{cases} n & \text{if } n \in V_G \\ \phi_{out}^{-1}(j) & \text{if } n = oc(H)[j] \text{ for some admissible } j \end{cases}$$

- Finally we define $link_G : P_G \uplus X \to E_G \uplus Y$. Take a port (n, i) with $n \in V_G$ and let $n := f(n_\bullet, n_0, \ldots, n_{h-1})$ be the proper assignment of n, then

  $$link_G((n, i)) = \begin{cases} n_i & \text{if } n_i \in E_G \\ Y[\phi_{out}^{-1}(j) - l] & \text{if } n_i = oc(H)[j] \text{ for some admissible } j \end{cases}$$

Consider instead a name $x = X[i]$ and let $\overline{x} = ic(H)[\phi_{in}(m+i)]$. The auxiliary assignment associated to $\overline{x}$ is $\overline{x} := n$. Thus

$$link_G(x) = \begin{cases} n & \text{if } n \in E_G \\ Y[\phi_{out}^{-1}(i) - l] & \text{if } n = oc(H)[j] \text{ for some } j \end{cases}$$

We can now present our first main correspondence result:

Theorem 1. *Shuffled support-equivalent bigraphs over a pure signature $\mathcal{K}$ are isomorphic to gs-graphs over $\mathcal{K}$ with name choices.*

The proof shows that $B[\![S[\![\cdot]\!]]\!]$ is the identity function on shuffled bigraphs and that $S[\![B[\![\cdot]\!]]\!]$ is the identity function on gs-graphs. Although not stressed here for space limitation, the transformations $S[\![\cdot]\!]$ and $B[\![\cdot]\!]$ preserve composition and tensor (see Appendix C), i.e., we can view bigraphs and gs-graphs not only as "essentially" equivalent formulations, but as "essentially" isomorphic algebras.

4 Characterising Binding Bigraphs

While in the pure case the correspondence can be worked out smoothly, the case of binding signatures is more challenging. At the signature level, the idea is just to consider operators of the form $K : \bullet\circ^h \to \bullet\circ^k$ for h the binding arity of K and k the free arity of K. Then we can straightforwardly define an injective transformation from (shuffled) binding bigraphs to gs-graphs as a suitable extension of the one in Section 3 (see Appendix D). The main difference is that now the class of gs-graphs freely generated by the signature may contain some elements that do not correspond to any valid binding graphs, because the scope discipline is not enforced by the free construction. Thus the transformation from binding bigraphs to gs-graphs is not surjective. Moreover the set of gs-graphs that are images of bigraphs is closed under monoidal product, but not under sequential composition. To see this, consider the gs-graphs in Fig. 4: the two gs-graphs on the left trivially respect the scope rule, but their sequential composition links h to the local port x of g despite h and g are siblings.

The main result we present here is the characterisation of "well-scoped" gs-graphs in terms of a relational type system in the style of relational separation logic [21]. We start by rephrasing the location principle for the scoping rule in the context of gs-graphs. We say that a name n is *bound* to location p if p and n appears together in the left hand side of some proper assignment in G (i.e. of the form $p, ..., n, ... := ...$). Sometimes we say just that n is bound, omitting the location p to which it is bound. If a name n is not bound then it is *free* (note that, for the purpose of this section, if n is assigned by $n := \nu$ then it is not bound and thus it is said free). If a location p is not assigned then we say it is *free*. We say that q *exploits* n if q and n appears together in the right hand side of some proper assignment in G (i.e. of the form $... := f(q, ..., n, ...)$).

Definition 19 (legal gs-graph). *A gs-graph G is* legal *iff for any p, q, n such that n is bound to p and q exploits n then $q \sqsubset_G^* p$, where $\sqsubset_G^*$ is the reflexive and transitive closure of $\sqsubset_G$.*

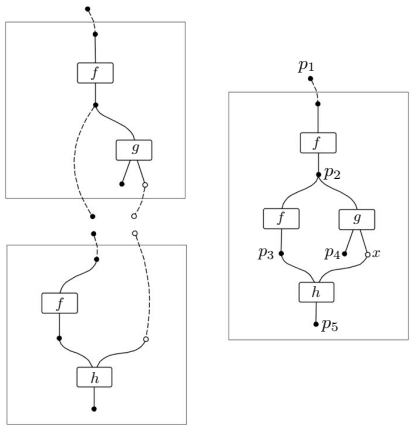

Fig. 4. An example of composition that does not respect the scope rule

As a special case, any "pure" gs-graphs is legal because pure signatures forbid the presence of names bound to locations. Our second main correspondence result is:

Theorem 2. *A gs-graph represents a binding bigraph iff it is legal.*

The proof goes by showing that the image of a binding bigraph via the transformation defined in Appendix D is a legal gs-graph (by contradiction, if it was not legal, then the scoping rule would have been violated) and then by giving a converse transformation from legal gs-graphs to binding bigraphs.

Example 5. Let us consider a binding signature with three operators $f : \bullet \to \bullet$, $g : \bullet\circ \to \bullet\circ^2$ and $h : \bullet \to \bullet\circ$. The gs-graph in Fig. 4 can be defined as $G = \{\ p_2 := f(p_1)\ ,\ p_3 := f(p_2)\ ,\ p_4, x := g(p_2)\ ,\ p_5 := h(p_3, x)\ ,\ !(p_4)\ ,\ !(p_5)\ \}$, which is not legal because p_3 exploits the node x (by the assignment $p_5 := h(p_3, x)$), which is bound to p_4 (by $p_4, x := g(p_2)$) and p_4 is not an ancestor of p_3 (i.e. $p_3 \not\sqsubseteq^*_G p_4$).

We can conveniently characterise the class of legal gs-graphs by exploiting an elegant type system. The typing relations we are interested in are of the form "p uses n" and "p misuses n". We need just three inference rules:

$$\frac{p \text{ free} \quad p \text{ uses } n}{p \text{ misuses } n} \qquad \frac{p, \ldots := f(q, \ldots, n, \ldots) \quad n \text{ bound}}{q \text{ uses } n}$$

$$\frac{p, n_1, \ldots, n_k := f(q, \ldots) \quad p \text{ uses } n \quad \forall i.\, n \neq n_i}{q \text{ uses } n}$$

Roughly the rules says that if q exploits n and n is bound to some other location, say p', then we must check that q be a descendant of p'. This task is accomplished by propagating "upward" the dependency through the location

hierarchy until either we discover that p' is an ancestor of q (in which case the propagation stops) or we reach the (free) root location p, in which case the scoping rule is violated and we assert that p misuses n. In the above example we have the typing relation $\{\, p_3 \text{ uses } x\,,\, p_2 \text{ uses } x\,,\, p_1 \text{ uses } x\,,\, p_1 \text{ misuses } x\,\}$.

Proposition 1. *A gs-graph is legal iff it induces an empty "misuses" relation.*

The proof is divided in two parts. First we show that if the "misuses" relation is not empty then the gs-graph is not legal. Conversely, we show that if a gs-graph is not legal, then the "misuses" relation is not empty.

Theorem 3. *The typing relation of $G_1 \otimes G_2$ is the union of the typing relations of G_1 and G_2. The typing relation of $G_1; G_2$ is a superset of the union of the typing relations of G_1 and of G_2.*

Corollary 1. *The parallel composition of two gs-graph is legal iff both are legal. If a non legal gs-graph is used in a sequential composition the result is non legal.*

Note that for computing the typing relation of $G_1; G_2$ it is enough to close the union of the typing relations of G_1 and G_2 w.r.t. the type inference rules (i.e. the reasoning is monotonic). The typing rules induce a straightforward quadratic algorithm for checking if a gs-graph is legal or not (the complexity is $O(B_G \cdot W_G)$ for B_G the number of $\bullet$ nodes in G and W_G the number of bound $\circ$ nodes in G).

5 Concluding Remarks

In conclusion, while bigraphs and gs-graphs are equally expressive, we claim that the presentation of gs-graphs in terms of sets of assignments combines the expressiveness of name links with the simpler and more standard algebraic structure of gs-monoidal theories. We believe that the relational type systems used above to check binding gs-graphs well-formedness may also be useful for establishing important properties of systems represented as gs-graphs.

A few observations are in place that deserves some future work. First, lean support equivalence over bigraphs abstract away from idle edges. Roughly, this corresponds to garbage collect restricted names that are not used and it is convenient for representing process calculi whenever the structural axiom $(\nu\ x)\mathbf{nil} = \mathbf{nil}$ is considered. The corresponding axiom for gs-monoidal theories would be $!_\circ\ ;\ \nu = id_\epsilon$, which has not been considered in this work because it is not part of the standard theory. At the level of abstract gs-graphs, this would correspond to require that the underlying multi-assignments G are such that whenever there is a name x such that G contains both $x := \nu$ and $!(x)$, then there is at least another assignment using x. This is a bit annoying because it requires some additional bookkeeping and cleansing when composing gs-graphs. Still, we are confident that our correspondence results will carry over smoothly to ones between amended gs-monoidal theories / gs-graphs and shuffled lean-support equivalent (binding) bigraphs.

Second, the research on bigraphs finds a main motivation in the reactive system approach mentioned in the introduction, which is based on the existence of so-called *relative push-out* (RPO) and *idem push-out* (IPO) in the category of bigraphs. RPOs/IPOs serve to distil the labelled transitions from the reduction rules and derive a bisimilarity equivalence that is guaranteed to be a congruence. Some preliminary investigation for extending the RPO approach to the case of term graphs has been reported in [7]. We conjecture that the variant of the reactive approach based on so-called *groupoidal RPOs* [20] can be applied to the category of shuffled bigraphs and hence of gs-graphs. Moreover, we would like to exploit the gs-monoidal structure and 2-categorical rewriting techniques, along the lines of [5], to define a reference theory of concurrent rewrites for bigraphs and gs-graphs, which is currently missing.

Third, the fact that legal gs-graphs do not compose may suggest that their interfaces miss some additional information. In fact, while we can always represent binding bigraphs as legal gs-graphs, the interfaces of gs-graphs remain the ones defined in the encoding for pure bigraphs and thus they are not able to pair names and the locations they are bound to. One possible solution could be to fix some convention from which the binding information can be automatically inferred. For example, we can assume that the names ($\circ$) listed in the interface are bound to the rightmost location ($\bullet$) appearing on their right, if any (and they are free otherwise) and use such hypotheses for checking that the gs-graph is legal or not. Yet, the information about the sharing of names between two or more location would get lost. We discarded this approach because, e.g., it would forbid the composition of legal gs-graphs with many arrows (i.e. symmetries like $\rho_{\circ,\bullet}$) that has no effect whatsoever on the essence of the gs-graph, but would change the hypotheses under which it has been tagged as legal. We plan to investigate this issue in more detail, as we think it has still many advantages over other proposals, like [10], which resort to the introduction of a much more powerful closed monoidal structure for the purpose.

Fourth, we would like to extend the comparison between binding bigraphs and legal gs-graphs to the algebra of graphs with nesting proposed in [1].

References

1. R. Bruni, A. Corradini, F. Gadducci, A. Lluch-Lafuente, and U. Montanari. On gs-monoidal theories for graphs with nesting. In *Graph Transformations and Model-Driven Engineering*, volume 5765 of *LNCS*, pages 59–86. Springer, 2010.
2. R. Bruni, F. Gadducci, and U. Montanari. Normal forms for algebras of connection. *Theor. Comput. Sci.*, 286(2):247–292, 2002.
3. R. Bruni, U. Montanari, and F. Rossi. An interactive semantics of logic programming. *TPLP*, 1(6):647–690, 2001.
4. L. Cardelli and A. D. Gordon. Mobile ambients. *Theor. Comput. Sci.*, 240(1):177–213, 2000.
5. A. Corradini and F. Gadducci. A 2-categorical presentation of term graph rewriting. In *CTCS'97*, volume 1290 of *LNCS*, pages 87–105. Springer, 1997.
6. A. Corradini and F. Gadducci. An algebraic presentation of term graphs, via gs-monoidal categories. *Applied Categorical Structures*, 7(4):299–331, 1999.

7. A. Corradini and F. Gadducci. On term graphs as an adhesive category. *Electr. Notes Theor. Comput. Sci.*, 127(5):43–56, 2005.
8. T. C. Damgaard and L. Birkedal. Axiomatizing binding bigraphs. *Nord. J. Comput.*, 13(1-2):58–77, 2006.
9. G. L. Ferrari and U. Montanari. Tile formats for located and mobile systems. *Inf. Comput.*, 156(1-2):173–235, 2000.
10. R. H. G. Garner, T. Hirschowitz, and A. Pardon. Variable binding, symmetric monoidal closed theories, and bigraphs. In *CONCUR'09*, volume 5710 of *LNCS*, pages 321–337. Springer, 2009.
11. M. Hyland and J. Power. Two-dimensional linear algebra. *Electr. Notes Theor. Comput. Sci.*, 44(1):227–240, 2001.
12. O. Jensen and R. Milner. Bigraphs and mobile processes (revised). Technical Report UCAM-CL-TR-580, University of Cambridge, 2004.
13. O. H. Jensen and R. Milner. Bigraphs and transitions. In *POPL*, pages 38–49, 2003.
14. S. MacLane. *Categories for the Working Mathematician*. Graduate Texts in Mathematics 5. Springer, 2nd edition, 1998.
15. J. Meseguer. Membership algebra as a logical framework for equational specification. In *WADT'97*, volume 1376 of *LNCS*, pages 18–61. Springer, 1997.
16. R. Milner. Bigraphical reactive systems. In *CONCUR'01*, volume 2154 of *LNCS*, pages 16–35. Springer, 2001.
17. R. Milner. Bigraphs whose names have multiple locality. Technical Report UCAM-CL-TR-603, University of Cambridge, 2004.
18. R. Milner. Bigraphs and their algebra. *Electr. Notes Theor. Comput. Sci.*, 209:5–19, 2008.
19. R. Milner. *The Space and Motion of Communicating Agents*. Cambridge University Press, 2009.
20. V. Sassone and P. Sobocinski. Locating reaction with 2-categories. *Theor. Comput. Sci.*, 333(1-2):297–327, 2005.
21. H. Yang. Relational separation logic. *Theor. Comput. Sci.*, 375(1-3):308–334, 2007.

A Composition of GS-Graphs

Atomic gs-graphs are defined as follows:

$$\begin{array}{rrll}
\text{(ops)} & f & \triangleq \{n'_1, \ldots, n'_{|u|} := f(n_1, \ldots n_{|v|})\} & \text{for any } f \in \Sigma_{u,v} \\
\text{(ids)} & id_s & \triangleq \{n_2 := n_1\} & \text{for } n_1, n_2 \text{ of sort } s \\
\text{(sym)} & \rho_{s,s'} & \triangleq \{n_3 := n_2\,,\, n_4 := n_1\} & \text{for } n_1, n_4 \text{ of sort } s' \text{ and } n_2, n_3 \text{ of sort } s \\
\text{(dup)} & \nabla_s & \triangleq \{n_2 := n_1\,,\, n_3 := n_1\} & \text{for } n_1, n_2, n_3 \text{ of sort } s \\
\text{(bang)} & !_s & \triangleq !(n) & \text{for } n \text{ of sort } s
\end{array}$$

Sequential and parallel composition are then defined below.

(seq): Let $G_1 : u \to v$ and $G_2 : v \to w$ such that $oc(G_1) = ic(G_2)$ and that no other names are shared between G_1 and G_2. Let A be the set of assignments in G_2 of the form $n := n'$ and let σ the corresponding name substitution. Their *sequential composition* is the gs-graph: $G_1; G_2 \triangleq (G_1\sigma) \cup (G_2 \backslash A) : u \to w$ with $ic(G_1; G_2) = ic(G_1)$ and $oc(G_1; G_2) = oc(G_2)$.

(par): Let $G_1 : u_1 \to v_1$ and $G_2 : v_2 \to w_2$ such that for every name n in G_1 and n' in G_2 we have $n \leq n'$. Their *parallel composition* is the gs-graph $G_1 \otimes G_2 : u_1u_2 \to v_1v_2 = G_1 \cup G_2$ for which we have $ic(G_1 \otimes G_2) = ic(G_1)ic(G_2)$ and $oc(G_1 \otimes G_2) = oc(G_1)oc(G_2)$.

Note that, if a composition is sound at the level of sorts, it is always possible to find suitable concrete gs-graphs in the equivalence classes that satisfy the constraints on names required to define their composition.

B More on (Pure) Bigraphs

B.1 Composition of Place Graphs

Let $P : m \to n$ and $Q : n \to l$ two concrete place graphs with $V_P \cap V_Q = \emptyset$, then their composition is defined as $Q \circ P \triangleq (V, ctrl, prnt) : m \to l$, where $V \triangleq V_P \uplus V_Q$ and

- for each $v \in V$ $ctrl(v) \triangleq \begin{cases} ctrl_P(v) \text{ if } v \in V_P \\ ctrl_Q(v) \text{ if } v \in V_Q \end{cases}$
- for each $v \in m \uplus V$ $prnt(v) \triangleq \begin{cases} prnt_P(v) \text{ if } v \in m \uplus V_P \text{ and } prnt_P(v) \in V_P \\ prnt_Q(i) \text{ if } v \in m \uplus V_P \text{ and } prnt_P(v) = i \in n \\ prnt_Q(v) \text{ if } v \in V_Q \end{cases}$

The identity place graph at m is $id_m \triangleq (\emptyset, \emptyset_{\mathcal{K}}, Id_m) : m \to m$ where Id_m is the identity function on the ordinal m.

The tensor product $\otimes$ on interfaces is the addition of ordinals and the unit object is 0. For $i \in \{0, 1\}$ let $P_i = (V_{P_i}, ctrl_{P_i}, prnt_{P_i}) : m_i \to n_i$ two place graphs having disjoint supports. Then

$$P_0 \otimes P_1 \triangleq (V_{P_0} \uplus V_{P_1}, ctrl_{P_0} \uplus ctrl_{P_1}, prnt_{P_0 \otimes P_1}) : m_0 + m_1 \to n_0 + n_1$$

where for each $j \in m_0 + m_1$

$$prnt_{P_0 \otimes P_1}(j) \triangleq \begin{cases} prnt_{P_0}(j) & \text{if } j < m_0 \\ prnt_{P_1}(j - m_0) & \text{if } j \geq m_0 \text{ and } prnt_{P_1}(j - m_0) \in V_{P_1} \\ prnt_{P_1}(j - m_0) + n_0 & \text{if } j \geq m_0 \text{ and } prnt_{P_1}(j - m_0) \in n_1 \end{cases}$$

and for each $v \in V_{P_0} \uplus V_{P_1}$

$$prnt_{P_0 \otimes P_1}(v) \triangleq \begin{cases} prnt_{P_0}(v) & \text{if } v \in V_{P_0} \\ prnt_{P_1}(v) & \text{if } v \in V_{P_1} \text{ and } prnt_{P_1}(v) \in V_{P_1} \\ prnt_{P_1}(v) + n_0 & \text{if } v \in V_{P_1} \text{ and } prnt_{P_1}(v) \in n_1 \end{cases}$$

Symmetries $\gamma_{m,n}$ have empty support and their effect is to swap sites:

$$\gamma_{m,n} \triangleq (\emptyset, \emptyset_{\mathcal{K}}, prnt(j) = \begin{cases} j + n \text{ if } j < m \\ j - m \text{ if } j \geq m \end{cases})$$

B.2 Composition of Link Graphs

Let $L : X \to Y$ and $M : Y \to Z$ be two link graphs such that $V_L \cap V_M = E_L \cap E_M = \emptyset$, then their composition is defined as: $M \circ L \triangleq (V, E, ctrl, link) : X \to Z$, where $V \triangleq V_L \uplus V_M$, $E \triangleq E_L \uplus E_M$ and

- for each $v \in V$ $ctrl(v) \triangleq \begin{cases} ctrl_L(v) & \text{if } v \in V_L \\ ctrl_M(v) & \text{if } v \in V_M \end{cases}$
- given a point $p \in X \uplus P_L \uplus P_M$ of $M \circ L$ then

$$link(p) \triangleq \begin{cases} link_L(p) & \text{if } p \in X \uplus P_L \text{ and } link_L(p) \in E_L \\ link_M(y) & \text{if } p \in X \uplus P_L \text{ and } link_L(p) = y \in Y \\ link_M(p) & \text{if } p \in P_M \end{cases}$$

The identity link graph at X is $id_X \triangleq (\emptyset, \emptyset, \emptyset_{\mathcal{K}}, Id_X) : X \to X$, with $Id_X : X \to X$ the identity function on the set X.

The product $\otimes$ is defined only on disjoint link graph interfaces, i.e. on disjoint sets of names, and it is roughly the disjoint set union. Suppose that for $i \in \{0, 1\}$, $L_i = (V_{L_i}, E_{L_i}, ctrl_{L_i}, link_{L_i}) : X_i \to Y_i$ are link graphs with disjoint supports and with $X_0 \cap X_1 = Y_0 \cap Y_1 = \emptyset$. Their product is:

$$L_0 \otimes L_1 \triangleq (V_{L_0} \uplus V_{L_1}, E_{L_0} \uplus E_{L_1}, ctrl_{L_0} \uplus ctrl_{L_1}, link_{L_0} \uplus link_{L_1}) : X_0 \uplus X_1 \to Y_0 \uplus Y_1$$

The unit object is the empty set $\emptyset$ and the symmetries $\gamma_{X,Y}$ are simply identities on $X \uplus Y$: $\gamma_{X_Y} \triangleq id_{X \uplus Y}$.

B.3 Composition of Bigraphs

Given two concrete bigraphs $G = (V_G, E_G, ctrl_G, prnt_G, link_G) : \langle m, X \rangle \to \langle n, Y \rangle$ and $H = \langle H^P, H^L \rangle : \langle l, Z \rangle$ with $V_G \cap V_H = E_G \cap E_H = \emptyset$, their composition is defined componentwise: $H \circ G \triangleq \langle H^P \circ G^P, H^L \circ G^L \rangle : \langle m, X \rangle \to \langle l, Z \rangle$

The identity bigraph at $\langle m, X \rangle$ is $id_{\langle m,X \rangle} \triangleq \langle id_m, id_X \rangle$.

Given two bigraph interfaces $\langle m, X \rangle$ and $\langle n, Y \rangle$ with $X \cap Y = \emptyset$, the product is defined componentwise: $\langle m, X \rangle \otimes \langle n, y \rangle = \langle m \otimes n, X \otimes Y \rangle$. The same happens on bigraphs: let for $i \in \{0, 1\}$ $G_i = \langle G^P, G^L \rangle : \langle m_i, X_i \rangle \to \langle n_i, Y_i \rangle$ be two bigraphs with disjoint supports and $X_0 \cap X_1 = Y_0 \cap Y_1 = \emptyset$, then

$$G_0 \otimes G_1 \triangleq \langle G_0^P \otimes G_1^P, G_0^L \otimes G_1^L \rangle : \langle m_0 + m_1, X_0 \uplus X_1 \rangle \to \langle n_0 + n_1, Y_0 \uplus Y_1 \rangle$$

The unit object ϵ is the pairing of the unit objects of the place graph and link graph products: $\epsilon = \langle 0, \emptyset \rangle$. Finally, for a pair of interfaces $\langle m, X \rangle$ and $\langle n, Y \rangle$ for which the product is defined, the symmetry is $\gamma_{\langle m,X \rangle, \langle n,Y \rangle} \triangleq \langle \gamma_{m,n}, \gamma_{X,Y} \rangle$.

B.4 Discrete Normal Form

Definition 20 (prime and discrete bigraph). *A bigraphical interface $\langle m, X \rangle$ is* prime *if $m = 1$ and we write it $\langle X \rangle$. A prime bigraph $G : m \to \langle X \rangle$ has no inner names and a prime outer interface. A bigraph D is* discrete *if it has no closed links, and its link map is bijective.*

Proposition 2 (discrete normal form). *Every bigraph $G : \langle m, X\rangle \to \langle n, Z\rangle$ can be expressed uniquely, up to a renaming on Y, as $G = (id_n \otimes \lambda) \circ D$, where $\lambda : Y \to Z$ is a linking and $D : \langle m, X\rangle \to \langle n, Y\rangle$ is discrete.*

Moreover, every discrete bigraph D may be factorised uniquely, up to a permutation of the sites of each factor, as $D = \alpha \otimes ((P_0 \otimes \cdots \otimes P_{n-1}) \circ \pi)$, with α a renaming, each P_i prime and discrete, and π a permutation of all the sites.

C Composition Preserving Transformations

Proposition 3 ($S[\![.]\!]$ preserves operations). *Suppose that $G : \langle m, X, \phi_{in}\rangle \to \langle n, Y, \psi\rangle$, and $G' : \langle n, Y, \psi\rangle \to \langle l, Z, \phi_{out}\rangle$ are two well formed shuffled bigraphs. We have that $S[\![G' \circ G]\!] = S[\![G]\!]; S[\![G']\!]$.*

Now consider $G_0 : \langle m_0, X_0, \phi_0\rangle \to \langle n_0, Y_0, \psi_0\rangle$ and $G_1 : \langle m_1, X_1, \phi_1\rangle \to \langle n_1, Y_1, \psi_1\rangle$ with $X_0 < X_1$ and $Y_0 < Y_1$, then $S[\![G_0 \otimes G_1]\!] = S[\![G_0]\!] \otimes S[\![G_1]\!]$.

Proposition 4 ($B[\![.]\!]$ preserves operations). *Let $H : (u, \sigma_u) \to (v, \sigma_v)$ and $H' : (v, \sigma_v) \to (w, \sigma_w)$ be gs-graphs with name choices, then $B[\![H; H']\!] = B[\![H']\!] \circ B[\![H]\!]$.*

Consider instead $H_0 : (u_0, \sigma_{u_0}) \to (v_0, \sigma_{v_0})$ and $H_1 : (u_1, \sigma_{u_1}) \to (v_1, \sigma_{v_1})$ gs-graphs with name choices such that $Im(\sigma_{u_0}) < Im(\sigma_{u_1})$ and $Im(\sigma_{v_0}) < Im(\sigma_{v_1})$. We have: $B[\![H_0 \otimes H_1]\!] = B[\![H_0]\!] \otimes B[\![H_1]\!]$.

D Transforming binding bigraphs in gs-graphs

The construction of the gs-graph representing a certain binding bigraph is not so different from that relative to pure bigraphs. In fact, as previously mentioned, we can not represent the locality relations in the context of gs-graphs and the transformation is forced to ignore them. Therefore we have to deal only with the added possibility for controls to declare names. Likewise the pure case, we work with gs-graphs equipped with name choices on the interfaces and with shuffled binding bigraphs. The latter are defined exactly like shuffled pure bigraphs (see Definition 18), except that in place of a pure bigraph we have clearly a binding bigraph. Let $G = (V_G, E_G, ctrl_G, prnt_G, link_G) : \langle m, loc_{in}, X, \phi_{in}\rangle \to \langle l, loc_{out}, Y, \phi_{out}\rangle$ be a shuffled binding bigraph on a signature $\mathcal{K}$ and denote with $P_G^{loc} \subseteq P_G$ the set of all its local ports. In particular given a node $v \in V_G$ such that its associated control $ctrl_G(v)$ has a positive internal arity, we call $(v, local_i)$ the $(i+1)^{th}$ local port declared by v.

In the gs-graph $H = S_{bind}[\![G]\!]$ the following names will appear:

$$\mathcal{N}_H^{\bullet} = V_G \uplus \{s_0, \ldots, s_{m-1}\} \uplus \{r_0, \ldots, r_{l-1}\}$$
$$\mathcal{N}_H^{\circ} = E_G \uplus \{x_0, \ldots, x_{|X|-1}\} \uplus \{y_0, \ldots, y_{|Y|-1}\}$$

Note that such sets of names are the same of those defined by the analogous transformation for the pure case. The difference is in the role played by the edges since in binding bigraphs they can be attached to local ports. In the

corresponding gs-graph such local edges are not assigned with the restriction operator ν, but within the proper assignment of the node that declares the local port to which it is linked. Since every edge can be attached to at most one local port (see Definition 14) we are guaranteed that each local edge is assigned exactly once. In the following we denote with E_G^{local} the set of all local edges belonging to the bigraph G.

As in Section 3 the overlined maps $\overline{prnt} : m \uplus V \to \mathcal{N}_H^\bullet$ and $\overline{link} : P_G \uplus X \to \mathcal{N}_H^\circ$ will help us in having a more concise representation of the assignments of H and we do not make any change to them. Nevertheless we report their definition here for a more quick reference.

$$\overline{prnt}(v) = \begin{cases} w \text{ if } prnt_G(v) = w \in V_G \\ r_i \text{ if } prnt_G(v) = i \in l \end{cases} \qquad \overline{link}(p) \triangleq \begin{cases} e \text{ if } link_G(p) = e \in E_G \\ y_i \text{ if } link_G(p) = Y[i] \end{cases}$$

Then we give the definitions of the assignments in H.

- $\forall v \in V_G$ with $ctrl_G(v) = f$ we add the assignment

  $$v\, \overline{link}(v, local_0)\ \ldots \overline{link}(v, local_{k-1}) := f(\overline{prnt}(v), \overline{link}(v, 0), \ldots, \overline{link}(v, k-1))$$

  where k and h are respectively the binding and the free arity of f. Note that since in binding bigraph a local port can be linked to an outer name, the link map and its overlined version applied on a local port return always an edge (that clearly is local).
- $\forall e \in E_G \backslash E_G^{local}$ we add $e := \nu$
- $\forall i \in m$ we add $s_i := \overline{prnt}(i)$
- $\forall x \in \{x_0, \ldots, x_{|X|-1}\}$ we add $x_i := \overline{link}(X[i])$

The inner and the outer connections are $\{s_0, \ldots, s_{m-1}\} \cup \{x_0, \ldots, x_{|X|-1}\}$ and $\{r_0, \ldots, r_{l-1}\} \cup \{y_0, \ldots, y_{|Y|-1}\}$ respectively and their order is obtained through the shuffle functions. In particular

$$ic(H) = (\overline{\phi}_{in}^{-1}(0), \overline{\phi}_{in}^{-1}(1), \ldots, \overline{\phi}_{in}^{-1}(m + |X| - 1))$$
$$oc(H) = (\overline{\phi}_{out}^{-1}(0), \overline{\phi}_{out}^{-1}(1), \ldots, \overline{\phi}_{out}^{-1}(l + |Y| - 1))$$

where $\overline{\phi}_{in}^{-1}$ and $\overline{\phi}_{out}^{-1}$ are defined as in Section 3.

In conclusion the name choices for the interfaces of $S_{bind}[\![G]\!]$ are $\sigma_{in}(i) \triangleq X[i]$ for $i \in |X|$ and $\sigma_{out}(i) \triangleq Y[i]$ for $i \in |Y|$.

We can now prove that the gs-graphs produced by $S_{bind}[\![.]\!]$ effectively represent binding bigraphs, or better, their bodies. We could not indeed encode the locality relation on the gs-graph interfaces, but we can show that the internal structure of a binding bigraph is faithfully represented.

For this purpose we abstract from the locality relation put on the interfaces and we identify all the binding bigraphs that differ only in these relations. We write $G \approx G'$ if G and G' are such two bigraphs and it follows immediately that $\approx$ is an equivalence relation. Furthermore, since $S_{bind}[\![.]\!]$ does not take into account

the locality relation, we have that $G \approx G'$ implies $S_{bind}[\![G]\!] = S_{bind}[\![G']\!]$ and therefore we can give a well-defined mapping on equivalence classes $S_{bind}\backslash_{\approx}[\![\cdot]\!]$ such that $S_{bind}\backslash_{\approx}[\![[G]]\!] = S_{bind}[\![G]\!]$, where $[G]$ denote the equivalence class of G.

Proposition 5. *The mapping $S_{bind}\backslash_{approx}[\![\cdot]\!]$ is injective.*

Unfortunately this mapping is not surjective. Next proposition guarantees that in the image of the mapping $S_{bind}[\![\cdot]\!]$ there are no unacceptable gs-graphs.

Proposition 6. *Let G be a binding bigraph. If in $S_{bind}[\![G]\!]$ there are an assignment $n\ldots := f(v,\ldots,e,\ldots)$ and an assignment $w\ldots e\ldots := g(\ldots)$ then $w \sqsubset^{+} v$.*

Finally we show that $S_{bind}[\![\cdot]\!]$ preserves the operations.

Proposition 7 ($S_{bind}[\![\cdot]\!]$ preserves operations). *Let $G : \langle m, loc_I, X, \phi_{in}\rangle \to \langle n, loc_J, Y, \psi\rangle$ and $G' : \langle n, loc_J, Y, \psi\rangle \to \langle l, loc_H, Z, \phi_{out}\rangle$, be shuffled binding bigraphs. Then $S_{bind}[\![G' \circ G]\!] = S_{bind}[\![G]\!]; S_{bind}[\![G']\!]$*

Suppose that $(G_0 : \langle m_0, loc_{I_0}, X_0, \phi_0\rangle \to \langle n_0, loc_{J_0}, Y_0, \psi_0\rangle)$ and $(G_1 : \langle m_1, loc_{I_1}, X_1, \phi_1\rangle \to \langle n_1, loc_{J_1}, Y_1, \psi_1\rangle)$ are shuffled binding bigraphs with $X_0 < X_1$ and $Y_0 < Y_1$. Then $S_{bind}[\![G_0 \otimes G_1]\!] = S_{bind}[\![G_0]\!] \otimes S_{bind}[\![G_1]\!]$