Binding bigraphs as symmetric monoidal closed theories

Tom Hirschowitz, Aurélien Pardon

Binding bigraphs as symmetric monoidal closed theories

Tom Hirschowitz[1] and Aurélien Pardon[2]

[1] CNRS, Université de Savoie `tom.hirschowitz@univ-savoie.fr`
[2] ENS Lyon `aurelien.pardon@ens-lyon.fr`

Abstract. We reconstruct Milner's [1] category of abstract binding bigraphs $\mathbf{Bbg}(\mathcal{K})$ over a signature $\mathcal{K}$ as the free (or initial) *symmetric monoidal closed* (SMC) category $S(\mathcal{T}_{\mathcal{K}})$ generated by a derived theory $\mathcal{T}_{\mathcal{K}}$. The morphisms of $S(\mathcal{T}_{\mathcal{K}})$ are essentially proof nets from the Intuitionistic Multiplicative fragment (IMLL) of Linear Logic [2].
Formally, we construct a faithful, essentially injective on objects functor $\mathbf{Bbg}(\mathcal{K}) \to S(\mathcal{T}_{\mathcal{K}})$, which is surjective on closed bigraphs (i.e., bigraphs without free names or sites). The functor is not full, which we view as a gain in modularity: we maintain the scoping discipline for whole programs (bound names never escape their scope) but allow more program fragments, including a large class of binding contexts, thanks to richer interfaces. Possible applications include bigraphical programming languages [3] and Rathke and Sobociński's derived labelled transition systems [4].

1 Overview

Milner's (binding) bigraphs [1] are a general framework for reasoning about distributed and concurrent programming languages, designed to encompass both the π-calculus [5] and the Ambient calculus [6]. We are here only concerned with bigraphical syntax: given what we call a *bigraphical signature* $\mathcal{K}$, Milner constructs a *pre-category*, and then a category $\mathbf{Bbg}(\mathcal{K})$, whose objects are *bigraphical interfaces*, and whose morphisms are bigraphs.

Its main features are (1) the presence of *relative pushouts* (RPOs) in the pre-category, which makes it well-behaved w.r.t. bisimulations, and that (2) in both the pre-category and the category, the so-called *structural* equations become equalities. Examples of the latter are, e.g., in π and Ambients, renaming of bound variables, associativity and commutativity of parallel composition, or scope extrusion for restricted names. Also, bigraphs follow a scoping discipline ensuring that, roughly, bound variables never escape their scope.

In this paper, we reconstruct bigraphs using standard algebraic tools. To explain this, let us quickly review the notion of a *many-sorted algebraic theory*, which is central in universal algebra. It is specified by first giving a *signature*—a set of sorts X and a set Σ of operations with arities—together with a set of *equations* over that signature. For example, the theory for monoids is specified by taking only one sort x, and operations $m : x \times x \to x$ and $e : 1 \to x$, together with the usual associativity and unitality equations. We may equally well view this signature as given by a graph

$$\begin{tikzcd}(x \times x) \arrow[r, "m"] & x & 1 \arrow[l, "e"']\end{tikzcd} \tag{1}$$

with vertices being the objects of the free category with (strict) finite products generated by X.

In this paper, we use the same kind of theories, but replacing from the start finite products with SMC structure. SMC structure is the categorical counterpart of IMLL [2,7]: through the Curry-Howard-Lambek correspondence, an SMC signature amounts to a set of IMLL axioms, and the free SMC category $S(\Sigma)$ over a signature Σ has as morphisms IMLL proofs under the corresponding axioms, modulo cut elimination. We use an isomorphic presentation of $S(\Sigma)$, essentially due to Trimble [7], in which morphisms are very much like IMLL *proof nets* [2]: they are kind of graphs, whose *correctness* is checked by (a mild generalisation of) the well-known Danos-Regnier criterion [8].

Here, we translate any bigraphical signature $\mathcal{K}$ into an SMC theory $\mathcal{T}_{\mathcal{K}}$, and consider the free SMC category $S(\mathcal{T}_{\mathcal{K}})$ generated by $\mathcal{T}_{\mathcal{K}}$ as an alternative category of bigraphs over $\mathcal{K}$. To compare $S(\mathcal{T}_{\mathcal{K}})$ with Milner's $\mathbf{Bbg}(\mathcal{K})$, we construct a faithful functor $\mathsf{T} : \mathbf{Bbg}(\mathcal{K}) \to S(\mathcal{T}_{\mathcal{K}})$. This functor is moreover essentially injective on objects (i.e., two objects with the same image are isomorphic), so that it is essentially an embedding.

However, our functor T is not full: even between bigraphical interfaces, $S(\mathcal{T}_{\mathcal{K}})$ contains morphisms which would be ill-scoped according to Milner's scope rule. Neither is T surjective on objects: $S(\mathcal{T}_{\mathcal{K}})$ has many more objects beyond bigraphical interfaces. This could be perceived as negative at first sight, but it actually represents a gain in modularity. Informally, even though our category is much more versatile, it still prevents bound variables to escape their scope. More formally, our functor is full on whole programs, i.e., bigraphs with no sites nor open names. Namely, it induces an isomorphism on closed terms, i.e., of hom-sets

$$S(\mathcal{T}_{\mathcal{K}})(I, t) \cong \mathbf{Bbg}(\mathcal{K})(I, t),$$

where I is the unit of tensor product, and t is a particular object representing terms[3]. So, our additional interfaces and morphisms allow just as many whole programs as Milner's, but more program fragments. Notably, it contains both the equivalent of terms and a kind of multi-hole, higher-order, binding contexts, all cohabiting happily.

In passing, our functor fully elucidates the status of so-called *edges* in bigraphs: we translate differently *free* edges (used for name restriction, much like ν in the π-calculus) and *bound* edges (used for linking so-called *binding ports* to their peers). In the former case, we translate the edge into a ν node (which may also be understood as representing the private name in question); in the latter case, we simply remove the edge, and rely on our use of directed graphs to represent the flow from the binding port to its peers.

[3] We cheat a little here, see the actual result Lemma 4.

Finally, our algebraic approach directly extends to what Debois [9] calls discrete sortings, which amount to a many-sorted variant of bigraphs. Given a set X of sorts, instead of only two sorts t and v for terms and variables (or names), one starts with sorts t_x and v_x for each $x \in X$, which directly allows to specify a typed signature.

Future work A possible application of our work is as an alternative representation for bigraphical programming languages [3]. For example, on the graphical side, the extensive litterature on efficient correctness criteria for proof nets applies directly. On the syntactic side, IMLL proofs provide an essentially algebraic representation, i.e., one avoiding the use of variable binding and the associated trickery [10,11,12].

Another possible application is as a handy foundation for Rathke and Sobociński's [4] work on deconstructing labelled transition systems, which involves second-order (binding) contexts.

As to future research directions, we here only handle abstract bigraphs, which do not have RPOs. We thus should generalise our approach to deal with *concrete* bigraphs, be it in the form of Milner's original pre-category or in Sassone and Sobociński's *G-categories* [13], and then try to construct the needed RPOs (or GRPOs).

Another natural research direction from this paper concerns the dynamics of bigraphs. Our hope is that Bruni et al.'s [14] very modular approach to dynamics may be revived, and work better with SMC structure than with cartesian closed structure. Specifically, with SMC structure, there is no duplication at the static level, which might simplify matters.

Related work The construction of the free SMC category generated by an SMC theory is essentially due to Trimble [7], followed by others [15,16,17]. The construction we use is a variant of Hughes' [16] construction, defined in our joint work with Richard Garner [18]. It was known that SMC (or cartesian closed) structure precisely represents various kinds of variable binding (see, e.g., Barber [19]). So we do not claim originality for the use of SMC structure to recover bigraphs.

Sassone and Sobociński [20] share our goal of categorically reconstructing bigraphs. They obtain very satisfactory results for pure bigraphs as a bicategory of cospans over a particular category of graphs. But their approach still has to scale up to deal with binding.

Damgaard and Birkedal [21] axiomatise the category of bigraphs as an equational theory over a term language with variable binding. Our work may be seen as an essentially algebraic counterpart of theirs (which relies on variable binding and the trickery evoked above). Moreover, they do not recognise the special status of bound edges, so our construction is not a mere reformulation of theirs in categorical terms.

Milner [22] and Debois [9] propose other extensions of Milner's original scope condition, the latter subsuming the former [9, Section 6.4]. Debois sees binding as a *sorting* over the category of pure bigraphs generated by $U(\mathcal{K})$, where U

forgets the binding information. His construction, starting from a 'no bound name escapes its scope' predicate, is reminiscent from Hyland and Tan's *double glueing* construction [15]. However, his construction is not known to satisfy any universal property, and an efficient (e.g., algebraic) representation of it still has to be found.

Finally, Grohmann and Miculan propose *directed bigraphs* [23] as a more flexible framework for bigraphs. Hildebrandt [24] proposes an algebraic approach to them, using *compact closed* categories instead of SMC categories. This line of work does not handle scope directly: one has to resort to a sorting in the above sense, with the same inconveniences.

Summary of contributions In view of the above, our contribution thus mainly lies in working out the details of the encoding of bigraphs as SMC theories, plus unraveling and simplifying the status of edges. Our task is made easier by Hughes' economic presentation of the free SMC category.

Furthermore, previous work using SMC structure as a representation for binding mostly lead to conservative extensions, i.e., full and faithful functors. Our work emphasises that Milner's *ad hoc* condition leaves room for generalisation, and provides a new, canonical condition, which benefits from efficient criteria from linear logical literature.

Structure of the paper Section 2 recalls the construction of the free SMC category generated by an SMC theory, including a specialisation to the case where a sort is equipped with a commutative monoid object structure. Section 3 reviews bigraphs, defining along the way our translation of bigraphical signatures. In Section 4, we construct our functor from bigraphs to the corresponding free SMC category, and show that it is an isomorphism on closed terms.

2 Symmetric monoidal closed theories

This section reviews SMC theories and their free models; see our note [18] for details. The constructions are essentially due to Trimble [7], but reworked using our extensions to Hughes' [16] presentation.

2.1 Signatures

We should start our overview with the definition of SMC categories: these are symmetric monoidal categories, i.e., categories with a tensor product $\otimes$ on objects and morphisms, symmetric in the sense that $A \otimes B \cong B \otimes A$, such that $(- \otimes A)$ has a right adjoint $(A \multimap -)$, for each A. We do not give further details, since we are interested in describing the free such category, which happens to be easier. Knowing that there is a category SMCCat of SMC categories and strictly structure-preserving functors should be enough to grasp the following.

As sketched in the introduction, a *signature* Σ is given by a set X of sorts, equipped with a graph whose vertices are IMLL formulae over X, as defined by the grammar:

$$A, B, \ldots \in \mathcal{F}(X) ::= x \mid I \mid A \otimes B \mid A \multimap B \qquad \text{(where } x \in X\text{)}.$$

We think of each edge $A \to B$ of the graph as specifying an operation of type $A \to B$. A morphism of signatures $(X, \Sigma) \to (Y, \Sigma')$ is a function $X \xrightarrow{f} Y$, equipped with a morphism of graphs, whose vertex component is "$\mathcal{F}(f)$", i.e., it sends any formula $A(x_1, \ldots, x_n)$ to $A(f(x_1), \ldots, f(x_n))$. This defines a category SMCSig of signatures.

There is then a forgetful functor $\mathsf{SMCCat} \xrightarrow{U} \mathsf{SMCSig}$ sending each SMC category $\mathcal{C}$ to the graph with

vertices: formulae in $\mathcal{F}(\mathrm{ob}(\mathcal{C}))$, and
edges $A \to B$**:** morphisms $[\![A]\!] \to [\![B]\!]$ in $\mathcal{C}$, where $[\![A]\!]$ is defined inductively to send each syntactic connective to the corresponding function on $\mathrm{ob}(\mathcal{C})$.

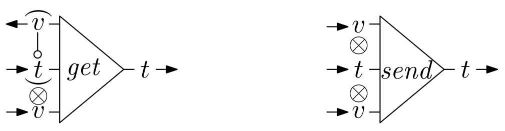
Trimble [7] constructs an SMC category $S(\Sigma)$ from any signature Σ, which extends to a functor $\mathsf{SMCSig} \xrightarrow{S} \mathsf{SMCCat}$, left adjoint to U: for any SMC category $\mathcal{C}$ and natural transformation $\Sigma \xrightarrow{f} U(\mathcal{C})$, there is a unique SMC functor $S(X) \xrightarrow{f^*} \mathcal{C}$ such that f decomposes as $X \xrightarrow{\eta} US(X) \xrightarrow{U(f^*)} U(\mathcal{C})$.

2.2 The free smc category

How does $S(\Sigma)$ look like? Its objects are IMLL formulae in $\mathcal{F}(X)$ and morphisms are kind of proof-nets with some *cells* representing operations. A morphism from A to B in $S(\Sigma)$ thus consists of

- a finite set $C = \{c, \ldots\}$ of cells labelled by operations $\alpha_c \to \beta_c$ in Σ,
- and *wires* connecting the *ports* together.

For example in the π-calculus, the operations *get* and *send* correspond to cells:



We always use the flat edge (or base) of the polygon to denote the domain. The orientation of a port corresponds to its sign, see below.

A port is a leaf occurrence (atomic or I) in A, B, or in some α_c, β_c. Equivalently, a port is a leaf occurrence in the formula

$$(A \otimes \bigotimes_{c \in C} (\alpha_c \multimap \beta_c)) \multimap B \qquad (2)$$

(or $A \multimap B$ when C is empty). In a formula, a port is *positive* when it lies to the left of an even number of $\multimap$'s, and *negative* otherwise[4]. The sign of a port in a morphism is its sign in the formula (2).

Wires are oriented, with sources the *negative* ports of the morphism, and targets in its *positive* ports. For each sort $x \in X$, the wires must induce a bijection between ports labelled x (which we call x ports). A negative I port can be wired to any positive port.

An example of morphism is presented in Fig. 1, where Σ has sorts $\{t, v\}$ and operations *get*, *send*, $v \xrightarrow{c} v \otimes v$, $I \xrightarrow{\nu} v$ and $t \otimes t \xrightarrow{p} t$.

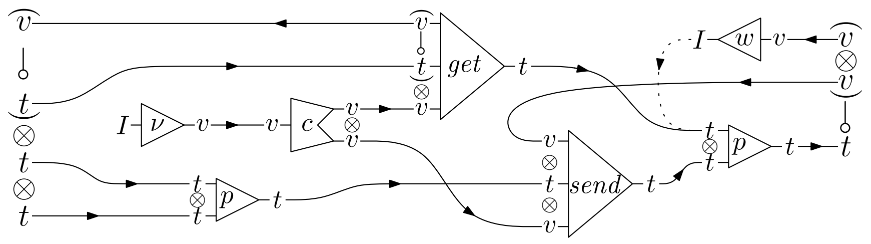


Fig. 1. A morphism of $S(\Sigma)$.

Correctness But, crucially, not all such graphs qualify as morphisms of $S(\Sigma)$: they have to satisfy a correctness criterion essentially due to Danos-Regnier [8], which goes as follows. The formula (2) may be written using the connectives of *classical* linear logic, defined by the grammar:

$$\begin{array}{rlllll} A, B, \ldots ::= x & \mid & 1 & \mid & A \otimes B \\ \mid \; x^{\perp} & \mid & \perp & \mid & A \parr B. \end{array}$$

We have removed $A \multimap B$, now encoded as $A^{\perp} \parr B$; some classical formulae are not expressible in IMLL, such as $\perp$, or $x \parr x$. The de Morgan dual $A^{\perp}$ of A is defined as usual.

A *switching* of a classical formula is its abstract syntax tree, where exactly one argument edge of each $\parr$ has been removed. A *switching* of a candidate morphism f is a graph obtained by gluing along ports the wires of f with a switching of the formula (2). The candidate is then *correct* iff all its switchings are acyclic and connected.

Equivalently, since (2) is ultimately a $\parr$, one may separately glue f with switchings of $A^{\perp}$, B, and each $\alpha_c \otimes \beta_c^{\perp}$. Notably, a cell of type $A \to x$ for some sort x is switched by connecting x to a switching of A.

[4] The sign of a port in A is directly apparent viewing A is a classical LL formula, see the next paragraph.

In Fig. 2 is pictured one switching (among 64) of the graph underlying the morphism in Fig. 1. The displayed connectives are those of the formula (2) translated into classical linear logic.

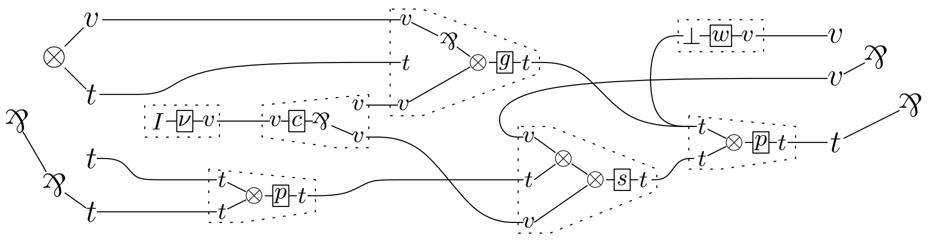


Fig. 2. A connected and acyclic switching of the morphism in Fig. 1.

Rewiring Finally, morphisms are quotiented by Trimble *rewiring*: a morphism *rewires* to another by changing the target of an edge from some negative I port, as soon as this preserves correctness. Rewiring is the smallest equivalence relation generated by this relation. In Fig. 1, the dotted wire starting from the negative I port can be rewired to any positive port because all switchings would remain trees.

2.3 Theories

That gives the construction for signatures. We now extend it to SMC theories: define a theory $\mathcal{T}$ to be given by a signature Σ, together with a set $E_{A,B}$ of equations between morphisms in $S(\Sigma)(A, B)$, for each IMLL formulae A, B. The free SMC category $S(\mathcal{T})$ generated by such a theory is defined in our note [18] to be the quotient of $S(\Sigma)$ by the equations. Constructing $S(\mathcal{T})$ graphically is more direct than could have been feared: we first define the binary predicate $f_1 \sim f_2$ relating two morphisms $C \overset{f_1,f_2}{\rightrightarrows} D$ in $S(\Sigma)$ as soon as each f_i decomposes as

$$C \xrightarrow{\cong} I \otimes C \xrightarrow{\ulcorner g_i \urcorner \otimes C} (A \multimap B) \otimes C \xrightarrow{f} D$$

with a common f, with $(g_1, g_2) \in E_{A,B}$, and where $\ulcorner g \urcorner$ is the currying of g. Then, we take the smallest generated equivalence relation, prove it stable under composition, and quotient $S(\Sigma)$ accordingly, which yields the free SMC category $S(\mathcal{T})$ generated by the theory $\mathcal{T} = (\Sigma, E)$.

Commutative monoid objects We finally slightly extend the results of our note [18] to better handle the special case of commutative monoids objects. This will be useful in our translation of bigraphs, where the sort t of terms has a commutative monoid structure given by parallel composition and $\mathbf{0}$. Assume a theory (Σ, E)

where a sort t is equipped with two operations m and e as in (1), with equations making it into a commutative monoid object (m is associative and commutative, e is its unit). Further assume that m and e do not occur in other equations.

Let Σ' be the result of removing the operations m and e in Σ. We define a relaxed version of our morphisms where each negative t port is connected to a positive one, but not necessarily bijectively. This defines a category isomorphic to $S(\Sigma)$, in which the operations m and e are built into the linking. The isomorphism is pictured in Fig. 3.

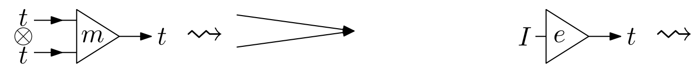


Fig. 3. Contracting m cells and deleting e cells.

3 Binding bigraphs and the translation of signatures

We now proceed to recall some definitions from Milner [1], along which we give our translation of bigraphical signatures $\mathcal{K}$ into SMC theories $\mathcal{T}_{\mathcal{K}}$. We then turn to our translation from the corresponding category of bigraphs to the free model $S(\mathcal{T}_{\mathcal{K}})$.

3.1 Signatures

Definition 1. *A* bigraphical (binding) signature *is a 4-uple* $(\mathcal{K}, B, F, \mathcal{A})$ *where* $\mathcal{K}$ *is a set of* controls*,* $B, F : \mathcal{K} \to \mathbb{N}$ *are maps providing a* binding *and a* free *arity for each control and* $\mathcal{A} \subseteq \mathcal{K}$ *is a set of* atomic *controls.*

We fix such a bigraphical signature $\mathcal{K}$ for the rest of the paper. This signature can be translated into a SMC signature $\Sigma_{\mathcal{K}}$ over two sorts $\{t, v\}$, standing for terms and variables (or names). It consists of the following *structural* operations, accounting for the built-in structure of bigraphs:

$$\begin{tikzcd} t \otimes t \arrow[r, "\mid"] & t & I \arrow[l, "\mathbf{0}"'] & & \\ & & & I \arrow[r, "\nu"] & v \\ v \otimes v & v \arrow[l, "c"'] \arrow[r, "w"] & I & & \end{tikzcd}$$

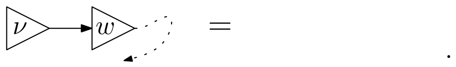
plus, for all controls k, a *logical* operation

$$(v^{\otimes B(k)} \multimap x) \otimes v^{\otimes F(k)} \xrightarrow{K_k} t$$

where $x = I$ if k is atomic and $x = t$ otherwise.

We call $\mathcal{T}_{\mathcal{K}}$ the theory consisting of the operations in $\Sigma_{\mathcal{K}}$, with the equations making

– $(t, |, \mathbf{0})$ into a commutative monoid object,
– (v, c, w) into a cocommutative comonoid object (c is coassociative, cocommutative, and w is its unit), and
– ν and w annihilate each other, as in



We now proceed to describe the category $\mathbf{Bbg}(\mathcal{K})$ of *abstract binding bigraphs* over $\mathcal{K}$, which we relate in Section 4 to the free model $S(\mathcal{T}_\mathcal{K})$ of $\mathcal{T}_\mathcal{K}$.

3.2 Interfaces

We assume an infinite and totally ordered set $\mathcal{X}$ of *names*.

Definition 2. *A* bigraphical (binding) interface *is a triple* (n, X, loc) *where* n *is a finite ordinal,* X *a finite set of names and* $loc : X \to n + \{\bot\}$ *a function called* locality map*.*

A name x is said *global* if $loc(x) = \bot$ and *local* or *located at* i when $loc(x) = i \in n$.

Bigraphical interfaces are the objects of the category $\mathbf{Bbg}(\mathcal{K})$. We define a function T from these objects to IMLL formulas, i.e., objects of $S(\mathcal{T}_\mathcal{K})$, by:

$$\mathsf{T} : (n, X, loc) \mapsto v^{\otimes n_g} \multimap \bigotimes_{i \in n} (v^{\otimes n_i} \multimap t) \tag{3}$$

where $n_g = |loc^{-1}(\bot)|$ and for all $i \in n$, $n_i = |loc^{-1}(i)|$. The ordering on $\mathcal{X}$ induces a bijection between X and v leaves in the formula.

In [22], Milner presents a slight generalisation of binding bigraphs, where names have multiple locality. Some interfaces cannot be simply translated into IMLL formulas as before, e.g., if x is located in 0 and 1 and y in 1 and 2, this dependency cannot be expressed directly in an IMLL formula.

3.3 Place graph

Let n and m be two finite ordinals.

Definition 3. *A* place graph $(V, ctrl, prnt) : n \to m$ *is a pair where:*

– V *is a finite set of* nodes*,*
– $ctrl : V \to \mathcal{K}$ *is a function called* control map *and*
– $prnt : n + V \to V + m$ *is an acyclic function called* parent map *whose image does not contain any atomic node.*

The ordinals n and m index respectively the *sites* and *roots*. A node is said *barren* if it has no preimage under the parent map (atomic nodes are thus barren).

The relation $\prec$ over sites, roots and nodes defined by:

$$x \prec y \iff \exists k > 0\,,\ prnt^k(x) = y$$

is a (strict) partial order. The maximal elements of $\prec$ are the roots; the minimal elements are the barren nodes (including atomic nodes) and the sites.

3.4 Link graph

Let X and Y be two finite sets of names.

Definition 4. *A* link graph $(V, E, \mathit{ctrl}, \mathit{link}) : X \to Y$ *is a tuple where:*

- V *is a finite set of* nodes*,*
- E *is a finite set of* edges*,*
- $\mathit{ctrl} : V \to \mathcal{K}$ *is a* control map *and*
- $\mathit{link} : P + X \to E + Y$ *is a function called the* link map

with P being the set of ports*, i.e., the coproduct of binding ports defined by* $P_B = \coprod_{v \in V} B(\mathit{ctrl}(v))$ *and free ports* $P_F = \coprod_{v \in V} F(\mathit{ctrl}(v))$*. Moreover,* link *must satisfy the* binding rule*:*

For all binding ports $p \in P_B$, $\mathit{link}(p) \notin Y$.

This binding rule is not mentionned in the original paper [1] about bigraphs whereas it is mandatory for the scoping discipline to be stable under composition (it is added in [22]). Alternatively, we can only require $\mathit{link}(p)$ to not be a global name of Y; the scope rule (defined below) handles the case of local names of Y.

We define the *binders* of our link graph to be the local names of Y (located at a root) and the binding ports (located at a node) P_B. Two distinct *points* (i.e., two elements of $P + X$) x and y are *peers* when $\mathit{link}(x) = \mathit{link}(y)$. An edge is *idle* when it has no preimage under the link map.

3.5 Abstract binding bigraphs

Let $U = (n, X, \mathit{loc})$ and $W = (m, Y, \mathit{loc}')$ be two bigraphical interfaces.

Definition 5. *A bigraph* $G = (V, E, \mathit{ctrl}, \mathit{prnt}, \mathit{link}) : U \to W$ *is a tuple where:*

- $(V, \mathit{ctrl}, \mathit{prnt}) : n \to m$ *is a place graph,*
- $(V, E, \mathit{ctrl}, \mathit{link}) : X \to Y$ *is a link graph,*
- G *satisfies the* scope rule*:*
  If p is a binder located at w, then each of its peers is located at some $w' \prec w$.

Example 1. An example of bigraph, roughly corresponding to the π-calculus context

$$\nu x(\bar{x}y.(\square_2 \mid \square_3) \mid x(z).\square_1)$$

where a global variable t is not used, is given in Fig. 4 using a representation from [1] and another from [24]. Binding (resp. free) names and ports are pictured by $\bullet$ (resp. $\circ$).

The binding rule ensures that no binding port p is peer of a name in Y, hence $\mathit{link}(p)$ has to be an edge. Moreover, by acyclicity of prnt, no two binding ports may be peers, hence edges are linked to at most one binding port. The set of edges may thus be decomposed into a set of *free* edges E_F (without binding

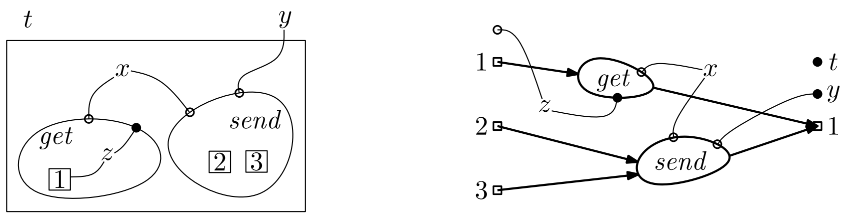


Fig. 4. Two bigraphical representations of the same π-term.

port) and a set of *bound* edges E_B in one-to-one correspondence with P_B by the link map: $E = E_F \uplus E_B \cong E_F + P_B$.

Finally, two bigraphs are *lean-support* equivalent when after discarding their idle edges, there is an isomorphism between their sets of nodes and edges preserving the structure.

Definition 6. *The category* $\mathbf{Bbg}(\mathcal{K})$ *of* abstract binding bigraphs over $\mathcal{K}$ *has bigraphical interfaces as objects and lean-support equivalence classes of bigraphs as morphisms.*

The composition of two bigraphs $U_1 \xrightarrow{G} U_2 \xrightarrow{G'} U_3$ is defined by taking the coproduct of their nodes, edges and control maps and the composition of parent and link maps (modulo some bijections on sets), forgetting the roots/sites from U_2. Acyclicity of the parent map, and the binding and scope rules are preserved by composition.

4 Translation

We now want to show how a binding bigraph $G = (V, E, \mathit{ctrl}, \mathit{prnt}, \mathit{link}) : U \to W$ over $\mathcal{K}$ can be translated into a morphism $\mathsf{T}(G) : \mathsf{T}(U) \to \mathsf{T}(W)$ in the free model $S(\mathcal{T}_\mathcal{K})$ of the SMC theory $\mathcal{T}_\mathcal{K}$. We have already defined T on objects in (3). Now, let $U = (n, X, \mathit{loc})$ and $W = (m, Y, \mathit{loc}')$. We will define the support C of $\mathsf{T}(G)$ as the disjoint union of:

- a *logical* support C containing a K_k cell for every node whose control is k and a ν cell for every free edge in E_F, and
- a *structural* support C' consisting of c and w cells, which we define below.

We then specify the graph $\mathsf{T}(G)$ for each sort in $\{t, v\}$ separately, and for I. For example, the image by T of the bigraph in Fig. 4 is the morphism in Fig. 1.

4.1 Places

First, since $(t, |, \mathbf{0})$ has a commutative monoid object structure, the representation of Section 2.3 applies: we just have to define a function from negative t

ports to positive ones. Now, for any set X labeled in formulae, denote by X_t^+ its set of positive t ports, and similarly for $X_{t,v,I}^{+,-}$. Now, considering each cell c to be labelled by the formula $\alpha_c \multimap \beta_c$, we have:

- $C_t^+ \cong V$, because each type of cell K_k has one positive t port,
- $C_t^- \cong V_{na} \hookrightarrow V$, where V_{na} is the set of non-atomic nodes, because there is one negative t port for each non-atomic cell,
- $\mathsf{T}(U)_t^+ \cong n$, because for each $i \in n$ there is a positive t port in $\mathsf{T}(U)$,
- similarly, $\mathsf{T}(W)_t^+ \cong m$, and finally
- $\mathsf{T}(W)_t^- \cong \mathsf{T}(U)_t^+ \cong \emptyset$.

Our morphism $\mathsf{T}(G)$ is thus defined on the sort t by the function f_t:

$$\begin{tikzcd}
\mathsf{T}(U)_t^+ + C_t^+ + \mathsf{T}(W)_t^- \arrow[r, "\cong"] \arrow[d, dashed, "f_t"'] & \mathsf{T}(U)_t^+ + C_t^+ \arrow[r, "\cong"] & n + V \arrow[d, "prnt"] \\
\mathsf{T}(U)_t^- + C_t^- + \mathsf{T}(W)_t^+ & \mathsf{T}(V)_t^+ + C_t^- \arrow[l, "\cong"'] & m + V_{na}. \arrow[l, "\cong"']
\end{tikzcd}$$

4.2 Links

The function f_v for v requires more work, and involves defining the structural support C'. Recall that the data is the function $link : P \uplus X \to E \uplus Y$.

We start with an informal description of f_v based on Fig. 5, in which bold arrows come from binders. First, we deal with points sent to edges. There are two kinds of edges.

First, we understand each free edge e as the creation of a fresh name, and each free point p in $P_F \uplus X$ sent to e as an occurrence of this free name. Accordingly, e is replaced by its ν cell in C, and each p becomes a v port in $\mathsf{T}(U)^- + C^-$. We hence link the v port of the ν cell to each corresponding p, through a tree of c and w cells, as depicted in the bottom row.

Second, we understand each bound edge e as an indirection to its binding port $p_0 \in P_B$, itself understood as a bound name. We further understand each free peer $p \in P_F \uplus X$ of p_0 as an occurrence of the bound name. Accordingly, we completely forget about e, p_0 becomes a v port in C^+, and each p becomes a v port in $C^- + \mathsf{T}(U)^-$, hence we link p_0 to each corresponding p, again through a tree of c and w cells.

Finally, points p not sent to an edge are sent to some name $y \in Y$. But each such p becomes a v port in $\mathsf{T}(U)^- + C^-$ and each such y becomes a v port in $\mathsf{T}(W)^-$, hence we link y to each p, again using c and w cells. This determines the structural support C', as well as f_v. Finally, for the I part f_I, each negative I port arises from a structural w cell. But in the above each cell is generated by *one* v port (the fresh name or the binder). In the former case, we may safely link our I port to any valid t port. In the latter, the binder occurs to the left of a $\multimap$, whose right-hand side is a t port, to which we safely link our I port.

More formally, observe from our translation of signatures and interfaces, plus the logical support C defined above, that:

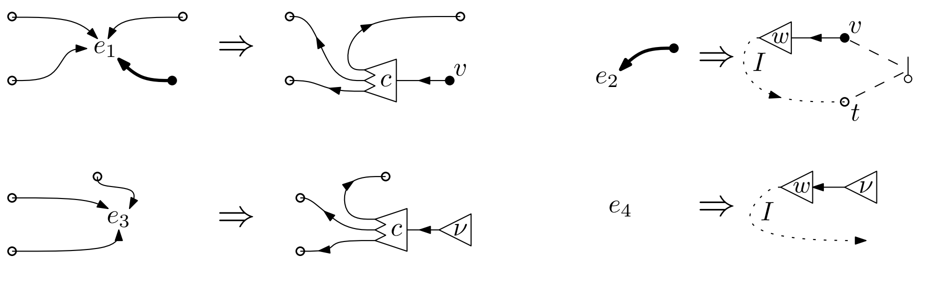


Fig. 5. Translation of *link*.

– each free edge in E_F corresponds to one ν cell, hence to one port in C_v^+,
– each binding port in P_B corresponds to one negative occurrence of v in the domain of some cell in C, hence to one port in C_v^+,
– each local name in Y corresponds one port in $\mathsf{T}(W)_v^-$.

Thus, we have an isomorphism $E_F + P_B + Y \cong C_v^+ + \mathsf{T}(W)_v^-$. Similarly, free points in $P_F + X$ correspond to ports in $C_v^- + \mathsf{T}(U)_v^-$, i.e., $P_F + X \cong C_v^- + \mathsf{T}(U)_v^-$.

We may thus define a first function *link*$'$ by:

$$\begin{tikzcd}
C_v^- + \mathsf{T}(U)_v^- \arrow[r, "\cong"] \arrow[d, dashed, "link'"'] & P_F + X \arrow[r, hook] & P_B + P_F + X \arrow[d, "link"] \\
C_v^+ + \mathsf{T}(W)_v^- & E_F + P_B + Y \arrow[l, "\cong"'] & E + Y. \arrow[l, "\cong"']
\end{tikzcd}$$

We then encode this function by a forest of c and w cells C' (as pictured in Fig. 6), to obtain a function $C_v^+ + C_v'^+ + \mathsf{T}(W)_v^- \xrightarrow{f_v} C_v^- + C_v'^- + \mathsf{T}(U)_v^-$, which qualifies as the v part of our morphism. The rest follows similarly.

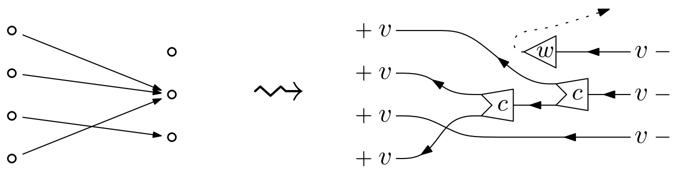


Fig. 6. Translation of a function using w and c cells.

This defines a function from bigraphs to candidate morphisms (respecting domain and codomain). We now show that it extends to a functor.

4.3 The functor

First, we prove that the image of a bigraph is correct, i.e., is a proper morphism.

Lemma 1. *All switchings of* $\mathsf{T}(G)$ *are connected.*

Proof. Essentially an induction over the place ordering $\prec$.

The following seems known [25]:

Lemma 2. *Any switching of a morphism in* $S(\mathcal{T}_\mathcal{K})$ *is acyclic iff it is connected.*

Proof (sketch). One proves by induction on the domain and codomain formulae that the graph induced by the switching has one more vertex than it has edges.

Lemma 3. *The map* $\mathsf{T} : \mathbf{Bbg}(\mathcal{K}) \to S(\mathcal{T}_\mathcal{K})$ *is a functor.*

Proof (sketch). The equations of $\mathcal{T}_\mathcal{K}$ defined in Section 3.1 ensure that T behaves well w.r.t. composition and lean-support equivalence.

One sees at once that T is not full. For example, the morphism in Fig. 7 has no preimage – any such preimage would violate the scope rule for bigraphs. This

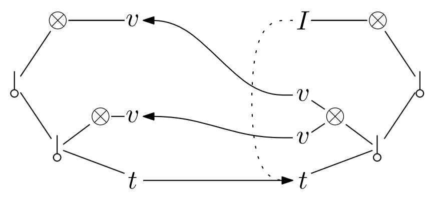


Fig. 7. A correct morphism violating the scope rule.

example reflects that it is not necessary to distinguish global and local variables in a bigraph with only one site. Nevertheless, the notion of scope is preserved by T because closed morphisms can actually be translated into bigraphs. In $\mathbf{Bbg}(\mathcal{K})$, define the interfaces $I = (0, \emptyset, \widehat{\emptyset})$ and $t = (1, \emptyset, \widehat{\emptyset})$.

Lemma 4. *The functor* T *induces an isomorphism* $S(\mathcal{T}_\mathcal{K})(I, t) \cong \mathbf{Bbg}(\mathcal{K})(I, t)$.

Proof (sketch). For any morphism $I \xrightarrow{f} t$, we construct a candidate bigraph, and observe that it vacuously satisfies the binding rule. Then, we proceed by contrapositive: assuming either that its parent map is cyclic or that it breaks the scope rule, we show that f was incorrect.

All in all, we have

Theorem 1. *The functor* $\mathsf{T} : \mathbf{Bbg}(\mathcal{K}) \to S(\mathcal{T}_\mathcal{K})$ *is faithful, essentially injective on objects, and surjective on* $S(\mathcal{T}_\mathcal{K})(I, t)$.

It is however not full and far from surjective on objects.

References

1. Jensen, O.H., Milner, R.: Bigraphs and mobile processes (revised). Technical Report TR580, University of Cambridge (2004)
2. Girard, J.Y.: Linear logic. Theoretical Computer Science **50** (1987) 1–102
3. Damgaard, T.C., Glenstrup, A.J., Birkedal, L., Milner, R.: An inductive characterization of matching in binding bigraphs. To appear (2008)
4. Rathke, J., Sobocinski, P.: Deconstructing behavioural theories of mobility. In: Fifth IFIP International Conference On Theoretical Computer Science - TCS '08, Springer (2008)
5. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes. Information and Computation **100**(1) (1992) 1–40
6. Cardelli, L., Gordon, A.: Mobile ambients. In Nivat, M., ed.: Foundations of Software Science and Computational Structures. Volume 1378 of Lecture Notes in Computer Science., Springer (1998) 140–155
7. Trimble, T.: Linear logic, bimodules, and full coherence for autonomous categories. PhD thesis, Rutgers University (1994)
8. Danos, V., Regnier, L.: The structure of multiplicatives. Archive for Mathematical Logic **28** (1989) 181–203
9. Debois, S.: Sortings & bigraphs. PhD thesis, IT University of Copenhagen (2008)
10. Pfenning, F., Elliott, C.: Higher-order abstract syntax. In: ACM SIGPLAN '88 Symposium on Language Design and Implementation, ACM (1988) 199–208
11. Gabbay, M.J., Pitts, A.M.: A new approach to abstract syntax involving binders. In: 14th Annual Symposium on Logic in Computer Science, IEEE Computer Society Press, Washington (1999) 214–224
12. Hofmann, M.: Semantical analysis of higher-order abstract syntax. In: LICS '99: Proc. 14th Annual IEEE Symposium on Logic in Computer Science, IEEE Computer Society (1999)
13. Sassone, V., Sobociński, P.: Deriving bisimulation congruences using 2-categories. Nordic Journal of Computing **10**(2) (2003) 163–183
14. Bruni, R., Montanari, U.: Cartesian closed double categories, their lambda-notation, and the pi-calculus. In: LICS '99: Proc. 14th Annual IEEE Symposium on Logic in Computer Science, IEEE Computer Society (1999) 246
15. Tan, A.: Full Completeness for Models of Linear Logic. PhD thesis, University of Cambridge (1997)
16. Hughes, D.J.D.: Simple free star-autonomous categories and full coherence. ArXiv Mathematics e-prints, math/0506521 (June 2005)
17. Lamarche, F., Strassburger, L.: From proof nets to the free *-autonomous category. Logical Methods in Computer Science **2**(4) (2006)
18. Garner, R.H.G., Hirschowitz, T., Pardon, A.: Graphical presentations of symmetric monoidal closed theories. CoRR **abs/0810.4420** (2008)
19. Barber, A., Gardner, P., Hasegawa, M., Plotkin, G.: From action calculi to linear logic. In: Annual Conference of the European Association for Computer Science Logic (CSL'97), Aarhus, August 1997, Selected Papers. Volume 1414 of Lecture Notes in Computer Science., Springer (1998) 78–97
20. Sassone, V., Sobociński, P.: Reactive systems over cospans. In: Logic in Computer Science, LiCS '05, IEEE Press (2005) 311–320
21. Damgaard, T., Birkedal, L.: Axiomatizing binding bigraphs. Nordic Journal of Computing **13**(1–2) (2006) 58–77

22. Milner, R.: Bigraphs whose names have multiple locality. Technical Report TR603, University of Cambridge (2004)
23. Grohmann, D., Miculan, M.: Directed bigraphs. Electr. Notes Theor. Comput. Sci. **173** (2007) 121–137
24. Hildebrandt, T.: Polarized and higher-order bigraphs from geometry of interaction. Talk at the Choco seminar, see `http://choco.pps.jussieu.fr/events`, Lyon (June 2008)
25. Soloviev, S.: Connectedness and acyclicity in IMLL proof nets (June 2008) Message to the TYPES forum.

A Proof of Lemma 1

Consider a switching of $\mathsf{T}(G)$.

Given a site or a node p, we denote by $\mathsf{T}(p)$ the negative t port corresponding to it in the switching. If p is a root, then $\mathsf{T}(p)$ denotes the positive t port of its image.

Free ports of a node p (resp. local names of a site p') have their image (a positive v port) connected to $\mathsf{T}(p)$ (resp. $\mathsf{T}(p')$) as shown in Fig. 8. Moreover, either one negative v port (corresponding to a binding port) or the positive t port of the cell p is connected to $\mathsf{T}(p)$ by the switched formula.

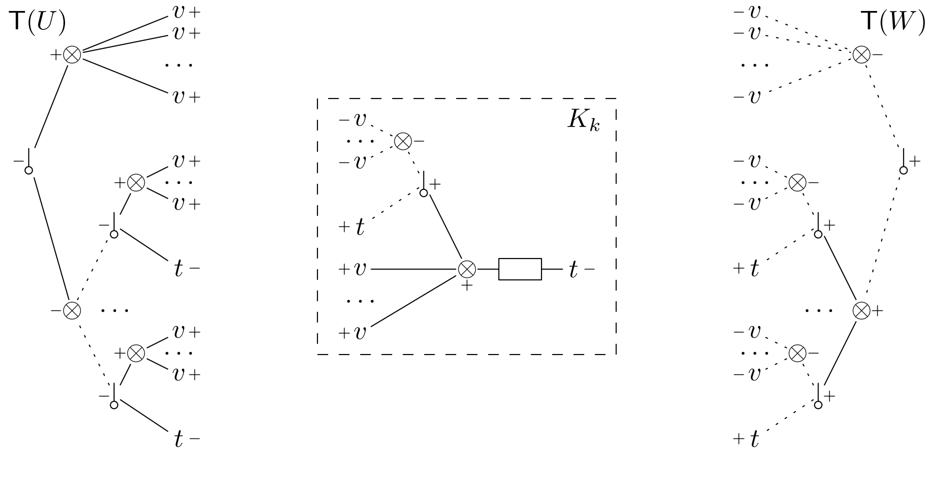


Fig. 8. Domain, codomain and a node of a switching.

We now prove by induction that all binding ports (located at a node or a root p) have their image connected to $\mathsf{T}(p)$. Let b be a binding port, and $\mathsf{T}(b)$ its image by T (a negative v port).

If b has no peers (this is necessarily the case if p is a barren node), then $\mathsf{T}(b)$ is connected to a w cell whose I port is connected to $\mathsf{T}(p)$.

If b has peers, then $\mathsf{T}(b)$ is connected, in the morphism, to their translations through a tree of c cells. But this tree is heavily switched and only connects $\mathsf{T}(b)$

to one positive v port f (whose preimage is) located, thanks to the scope rule, to a site or a node $p' \prec p$.

By induction f is connected to $\mathsf{T}(p')$ and $\mathsf{T}(p')$ is connected to $\mathsf{T}(p)$ through the (unswitched) parent map. Indeed, the parent map connects the t ports of cells between p' and p, and these cells have their t ports connected thanks to the induction hypothesis. The port $\mathsf{T}(b)$ and $\mathsf{T}(p)$ are thus connected.

Finally, we remark that:

- roots are connected to each other in the codomain's formula (by their t or v ports, see Fig. 8),
- global variables of the domain are connected to a site (by the domain's formula, see Fig. 8) and
- remaining negative v ports (global variable of the codomain and ν cells) are connected to the other positive ports by a switched tree of c cells or a w cell.

We conclude that all ports of our switching are connected.

B Proof of Lemma 4

Consider any $f : I \to t$. We have $\mathsf{T}(I) = (I \multimap I) \cong I$ and $\mathsf{T}(t) = I \multimap (I \multimap t) \cong t$, which justifies our "induces" above. We now define $G = (V, E, \mathit{ctrl}, \mathit{prnt}, \mathit{link}) : I \to t$ such that $\mathsf{T}(G) = f$.

Let the set of nodes V be the set of logical cells in f; the control map ctrl sends each K_k cell to $k \in \mathcal{K}$.

The set of edges is the coproduct of binding v ports in the support of f and of ν cells (where a v port is binding when it occurs to the left of a $\multimap$, e.g., a cell of type $((v \otimes v) \multimap t) \otimes v \to t$ has two binding ports).

The parent map $\mathit{prnt} : 0 + V \to 1 + V$ is exactly the restriction of f to t ports. The link map $\mathit{link} : P_B + P_F + \emptyset \to E + \emptyset$ is obtained from the restriction of f to v ports as follows. From any v port p, following the tree of contractions towards its root leads to a maximal positive v port in the support, which may be either a port from a ν cell, or a binding port of a logical cell. In each case, there is a corresponding edge e_p. Our link map sends each port p to e_p. Since in each tree there is only one root, the binding rule is respected.

We then prove that G is correct. Suppose that the parent map contains a cycle, then any switching where, for all cells of the cycle, the two t ports are connected contains this cycle. Suppose that the scope rule is not satisfied for a binder p and one of its peers p'. Then, in f, p is the root of a contraction tree with p' as a leaf: among the switchings connecting them, choose again one that connects both t ports of each logical cell: every logical cell then has a path to the root r (the t port in the codomain), which forms a cycle involving p, p', and r, hence contradicting correctness of f. The binding rule is automatically satisfied because the codomain has no name. An atomic node has no antecedent in the parent map because the corresponding cell in f has no positive t port.