# Typed Polyadic Pi-calculus in Bigraphs

Vladimiro Sassone

ECS, University of Southampton

joint work with M. Bundgaard (IT Copenhagen)

PPDP'06, Venice 10 July 2006

# Main operational models of mobile systems

- **NAME MOBILITY** (typically, pi-calculus)

$$(\nu z)(\overline{x}\langle z\rangle.P \mid Q) \mid x(y).R \quad \longrightarrow \quad (\nu z)(P \mid Q \mid R[z/y])$$

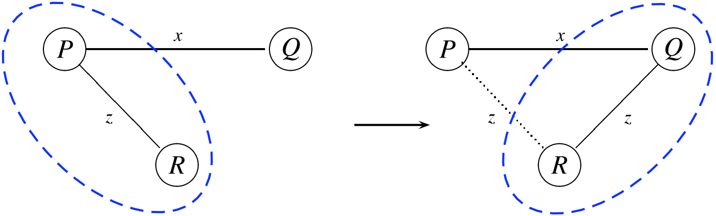
- **PROCESS MOBILITY** (typically, distributed pi-calculus)

$$\ell_0[\,\mathbf{goto}\ \ell_1.P \mid Q\,] \mid \ell_1[\,R\,] \quad \longrightarrow \quad \ell_0[\,Q\,] \mid \ell_1[\,P \mid R\,]$$

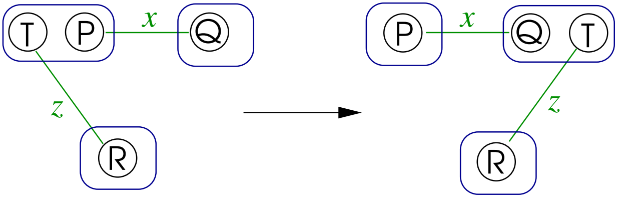
- **AMBIENT MOBILITY** (typically, ambient calculus)

$$n[\,\mathbf{in\ m}.P \mid Q\,] \mid m[\,R\,] \quad \longrightarrow \quad m[\,n[\,P \mid Q\,] \mid R\,]$$
$$m[\,n[\,\mathbf{out\ m}.P \mid Q\,] \mid R\,] \quad \longrightarrow \quad n[\,P \mid Q\,] \mid m[\,R\,]$$

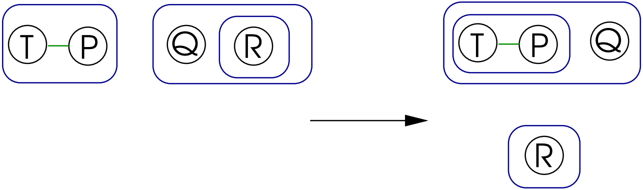
# The essence of name mobility



# The essence of process mobility

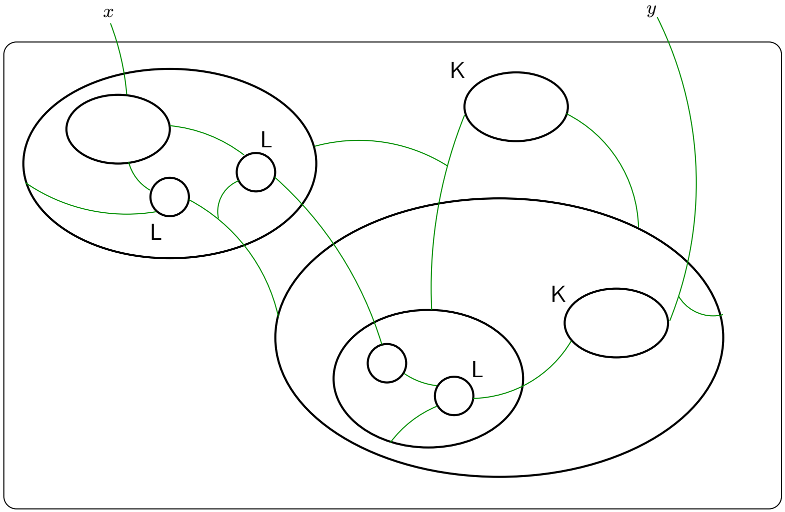


# The essence of ambient mobility

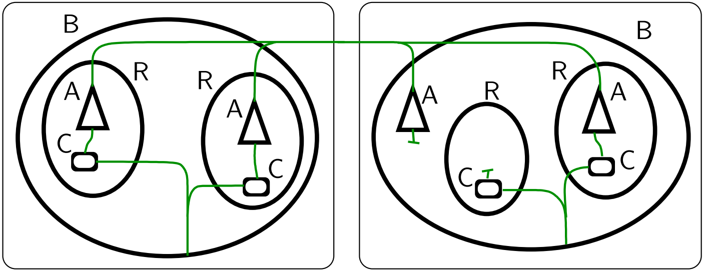


# Bigraphs as a unifying model

## Overlapping **placing** and **linking** structure



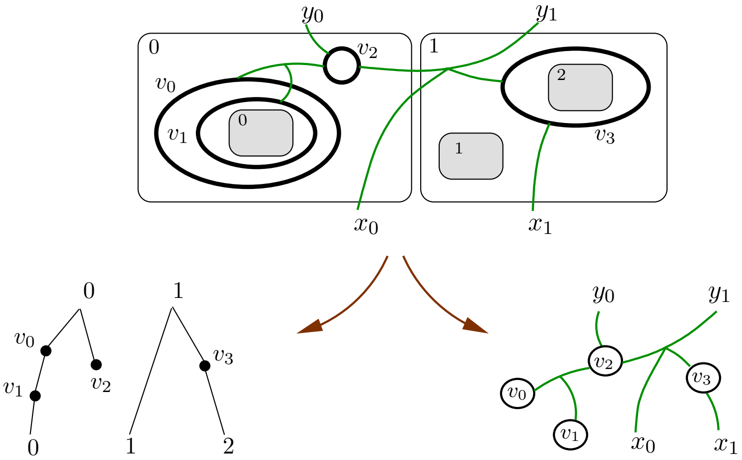
# A slightly more suggestive example




A - an agent    B - a building
C - a computer    R - a room

# Pure bigraphs

$G : \langle 3, \{x_0, x_1\} \rangle \to \langle 2, \{y_0, y_1\} \rangle$




$G^P : 3 \to 2$

$G^L : \{x_0, x_1\} \to \{y_0, y_1\}$

# Pure bigraphs, formally (1)

**Definition (Signature of controls $\mathcal{K}$)**

$K \in \mathcal{K}$ has an **arity** $ar(K)$, its number of ports.
A control can be **atomic**, and then not allowed to contain further structure. Non-atomic controls can be **active** or **passive**.

**Definition (Place graph over $\mathcal{K}$)**

$G^{\mathrm{P}} = (V, ctrl, prnt) : m \to n$ with inner width (sites) $m$ and outer width (roots) $n$ consisting of

- a **control map** $ctrl : V \to \mathcal{K}$; which assigns controls to nodes;
- an acyclic **parent map** $prnt : m \uplus V \to V \uplus n$;
  (atomic nodes may not be parents).

# Pure bigraphs, formally (1)

**Definition (SIGNATURE OF CONTROLS $\mathcal{K}$)**

$K \in \mathcal{K}$ has an **arity** $ar(K)$, its number of ports.
A control can be **atomic**, and then not allowed to contain further structure. Non-atomic controls can be **active** or **passive**.

**Definition (PLACE GRAPH OVER $\mathcal{K}$)**

$G^{\mathsf{P}} = (V, ctrl, prnt) : m \to n$ with inner width (sites) $m$ and outer width (roots) $n$ consisting of

- a **control map** $ctrl : V \to \mathcal{K}$; which assigns controls to nodes;
- an acyclic **parent map** $prnt : m \uplus V \to V \uplus n$; (atomic nodes may not be parents).

# Pure bigraphs, formally (2)

## Definition (LINK GRAPH OVER $\mathcal{K}$)

$G^L = (V, E, ctrl, link) \colon X \to Y$ with inner names $X$ and outer names $Y$ consisting of

- a **control map** $ctrl \colon V \to \mathcal{K}$;
- a finite set of **edges** $E$;
- a **link map** $link \colon X \uplus P \to E \uplus Y$ mapping 'points' to 'links', where $P \stackrel{\text{def}}{=} \sum_{v \in V} ar(ctrl(v))$ are called the **ports** of $G^L$.

## Terminology

A link is **idle** if it has no preimage under the link map; **open** if it is an (outer) name; **closed** if it is an edge.

A link graph is **lean** if it has no idle edges.

A point is **open** if its link is open, it is **closed** otherwise.

# Pure bigraphs, formally (2)

## Definition (LINK GRAPH OVER $\mathcal{K}$)

$G^{\mathsf{L}} = (V, E, ctrl, link) \colon X \to Y$ with inner names $X$ and outer names $Y$ consisting of

- a **control map** $ctrl \colon V \to \mathcal{K}$;
- a finite set of **edges** $E$;
- a **link map** $link \colon X \uplus P \to E \uplus Y$ mapping 'points' to 'links', where $P \stackrel{\text{def}}{=} \sum_{v \in V} ar(ctrl(v))$ are called the **ports** of $G^{\mathsf{L}}$.

## Terminology

A link is **idle** if it has no preimage under the link map; **open** if it is an (outer) name; **closed** if it is an edge.

A link graph is **lean** if it has no idle edges.

A point is **open** if its link is open, it is **closed** otherwise.

# Pure bigraphs, formally (3)

**Definition (PURE BIGRAPHS)**

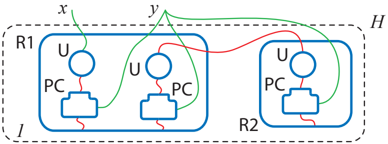
The superimposition of a place and a link graph sharing nodes and control map. Namely,

$$G = (V, E, ctrl, prnt, link) \colon \langle m, X\rangle \to \langle n, Y\rangle$$

where $G^{P} = (V, ctrl, prnt) \colon m \to n$ is a place graph,
and $G^{L} = (V, E, ctrl, link) \colon X \to Y$ is a link graph.

# Bigraphs as a $(\otimes, \circ)$-algebra

$$H = G \circ (F_1 \otimes F_2) : \epsilon \to \langle 1, \{x, y\} \rangle$$



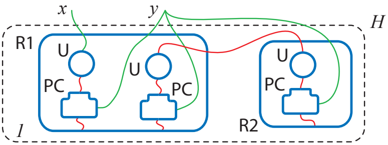

$$F_1 : \epsilon \to \langle 1, \{x, y\} \rangle \qquad F_2 : \epsilon \to \langle 1, \{z, v, w\} \rangle$$

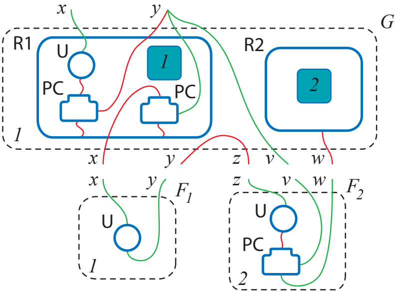
$$\epsilon \xrightarrow{F_1 \otimes F_2} \langle 2, \{x, y, z, v, w\} \rangle \xrightarrow{G} \langle 1, \{x, y\} \rangle$$

Essentially a symmetric monoidal category (more later).

# Bigraphs as a $(\otimes, \circ)$-algebra

$$H = G \circ (F_1 \otimes F_2) : \epsilon \to \langle 1, \{x, y\} \rangle$$



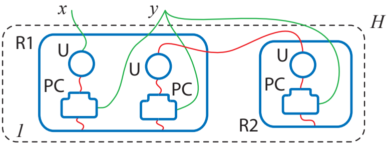

$F_1 : \epsilon \to \langle 1, \{x, y\} \rangle \qquad F_2 : \epsilon \to \langle 1, \{z, v, w\} \rangle$

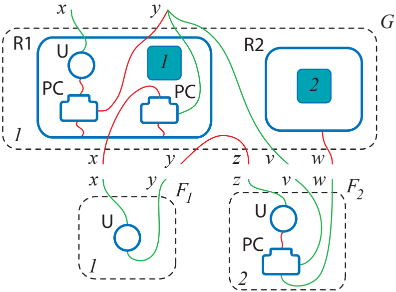
$$\epsilon \xrightarrow{F_1 \otimes F_2} \langle 2, \{x, y, z, v, w\} \rangle \xrightarrow{G} \langle 1, \{x, y\} \rangle$$

Essentially a symmetric monoidal category (more later).

# Bigraphs as a $(\otimes, \circ)$-algebra

$$H = G \circ (F_1 \otimes F_2) : \epsilon \rightarrow \langle 1, \{x, y\} \rangle$$



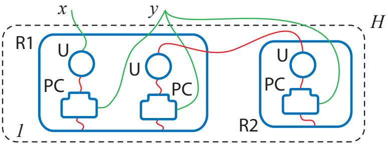

$$F_1 : \epsilon \rightarrow \langle 1, \{x, y\} \rangle \qquad F_2 : \epsilon \rightarrow \langle 1, \{z, v, w\} \rangle$$

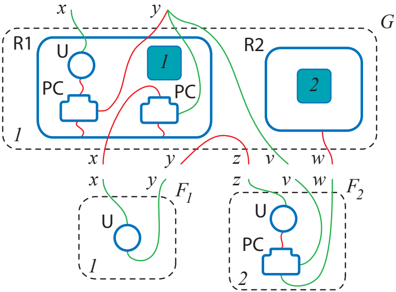
$$\epsilon \xrightarrow{F_1 \otimes F_2} \langle 2, \{x, y, z, v, w\} \rangle \xrightarrow{G} \langle 1, \{x, y\} \rangle$$

Essentially a symmetric monoidal category (more later).

# Bigraphs as a $(\otimes, \circ)$-algebra

$$H = G \circ (F_1 \otimes F_2) : \epsilon \rightarrow \langle 1, \{x, y\} \rangle$$



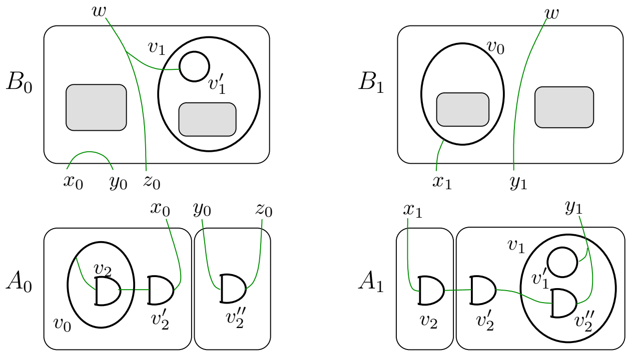




$$F_1 : \epsilon \rightarrow \langle 1, \{x, y\} \rangle \qquad F_2 : \epsilon \rightarrow \langle 1, \{z, v, w\} \rangle$$

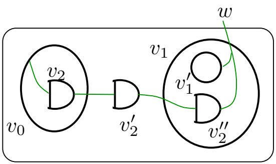
$$\epsilon \xrightarrow{F_1 \otimes F_2} \langle 2, \{x, y, z, v, w\} \rangle \xrightarrow{G} \langle 1, \{x, y\} \rangle$$

Essentially a symmetric monoidal category (more later).

# A second example of $(\otimes, \circ)$-composition




$B_0 \circ A_0 = B_1 \circ A_1$



# The s-category of pure bigraphs

Pure bigraphs are the arrows of a **s-category** $\acute{}\mathrm{B}\textsc{ig}(\mathcal{K})$.

They fail to be a category as both $\circ$ and $\otimes$ are partial. Oh my!

- $\circ$ is partial because of the underlying concrete sets of nodes; s-categories make provisions in that respect.
- $\otimes$ is partial because of the sets of names; as names cannot be taken up to iso, this is an intrinsic feature. It is not so bad.

**Abstract bigraphs** are the quotient classes of 'lean-support' equivalence $\risingdotseq$, where $G \risingdotseq H$ if they are isomorphic after discarding all idle edges.

$\mathrm{B}\textsc{ig}(\mathcal{K})$ is the category of abstract pure bigraphs, that is symmetric partial monoidal. We only focus on concrete bigraphs; under suitable conditions all can be transferred to abstract ones via the quotient functor $[\![\cdot]\!] : \acute{}\mathrm{B}\textsc{ig}(\mathcal{K}) \to \mathrm{B}\textsc{ig}(\mathcal{K})$.

# The s-category of pure bigraphs

Pure bigraphs are the arrows of a **s-category** $\acute{}\mathrm{BIG}(\mathcal{K})$.

They fail to be a category as both $\circ$ and $\otimes$ are partial. Oh my!

- $\circ$ is partial because of the underlying concrete sets of nodes; s-categories make provisions in that respect.
- $\otimes$ is partial because of the sets of names; as names cannot be taken up to iso, this is an intrinsic feature. It is not so bad.

**Abstract bigraphs** are the quotient classes of 'lean-support' equivalence $\stackrel{\frown}{=}$, where $G \stackrel{\frown}{=} H$ if they are isomorphic after discarding all idle edges.

$\mathrm{BIG}(\mathcal{K})$ is the category of abstract pure bigraphs, that is symmetric partial monoidal. We only focus on concrete bigraphs; under suitable conditions all can be transferred to abstract ones via the quotient functor $[\![\cdot]\!] : \acute{}\mathrm{BIG}(\mathcal{K}) \rightarrow \mathrm{BIG}(\mathcal{K})$.

# The s-category of pure bigraphs

Pure bigraphs are the arrows of a **s-category** $\acute{\mathrm{B}}\mathrm{IG}(\mathcal{K})$.

They fail to be a category as both $\circ$ and $\otimes$ are partial. Oh my!

- $\circ$ is partial because of the underlying concrete sets of nodes; s-categories make provisions in that respect.
- $\otimes$ is partial because of the sets of names; as names cannot be taken up to iso, this is an intrinsic feature. It is not so bad.

**Abstract bigraphs** are the quotient classes of 'lean-support' equivalence $\risingdotseq$, where $G \risingdotseq H$ if they are isomorphic after discarding all idle edges.

$\mathrm{B}\mathrm{IG}(\mathcal{K})$ is the category of abstract pure bigraphs, that is symmetric partial monoidal. We only focus on concrete bigraphs; under suitable conditions all can be transferred to abstract ones via the quotient functor $[\![\cdot]\!] : \acute{\mathrm{B}}\mathrm{IG}(\mathcal{K}) \to \mathrm{B}\mathrm{IG}(\mathcal{K})$.

# The s-category of pure bigraphs

Pure bigraphs are the arrows of a **s-category** $\acute{}\mathrm{BIG}(\mathcal{K})$.

They fail to be a category as both $\circ$ and $\otimes$ are partial. Oh my!

- $\circ$ is partial because of the underlying concrete sets of nodes; s-categories make provisions in that respect.
- $\otimes$ is partial because of the sets of names; as names cannot be taken up to iso, this is an intrinsic feature. It is not so bad.

**Abstract bigraphs** are the quotient classes of 'lean-support' equivalence $\stackrel{\circ}{=}$, where $G \stackrel{\circ}{=} H$ if they are isomorphic after discarding all idle edges.

$\mathrm{BIG}(\mathcal{K})$ is the category of abstract pure bigraphs, that is symmetric partial monoidal. We only focus on concrete bigraphs; under suitable conditions all can be transferred to abstract ones via the quotient functor $[\![\cdot]\!] : \acute{}\mathrm{BIG}(\mathcal{K}) \to \mathrm{BIG}(\mathcal{K})$.

# The s-category of pure bigraphs

Pure bigraphs are the arrows of a **s-category** $\acute{}\mathrm{BIG}(\mathcal{K})$.

They fail to be a category as both $\circ$ and $\otimes$ are partial. Oh my!

- $\circ$ is partial because of the underlying concrete sets of nodes; s-categories make provisions in that respect.
- $\otimes$ is partial because of the sets of names; as names cannot be taken up to iso, this is an intrinsic feature. It is not so bad.

**Abstract bigraphs** are the quotient classes of 'lean-support' equivalence $\doteqdot$, where $G \doteqdot H$ if they are isomorphic after discarding all idle edges.

$\mathrm{BIG}(\mathcal{K})$ is the category of abstract pure bigraphs, that is symmetric partial monoidal. We only focus on concrete bigraphs; under suitable conditions all can be transferred to abstract ones via the quotient functor $[\![\cdot]\!] : \acute{}\mathrm{BIG}(\mathcal{K}) \to \mathrm{BIG}(\mathcal{K})$.

# Merging roots and outer names in products

Two derived operators allow to merge outer names and roots

- **PARALLEL PRODUCT:** $G_0 \parallel G_1$: like $\otimes$ with merge of outer names;

  It works by first making all the outer names disjoint, then composing with $\otimes$, and finally renaming names as originally in the resulting bigraph.

- **PRIME PRODUCT:** $G_0 \mid G_1$: like $\parallel$ with merge of roots in the resulting bigraph.

# Merging roots and outer names in products

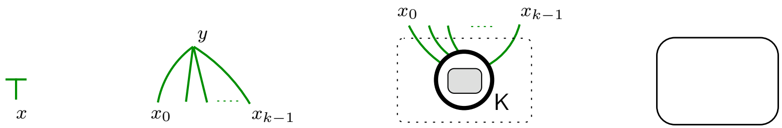
Two derived operators allow to merge outer names and roots

- **PARALLEL PRODUCT:** $G_0 \parallel G_1$: like $\otimes$ with merge of outer names;

  It works by first making all the outer names disjoint, then composing with $\otimes$, and finally renaming names as originally in the resulting bigraph.

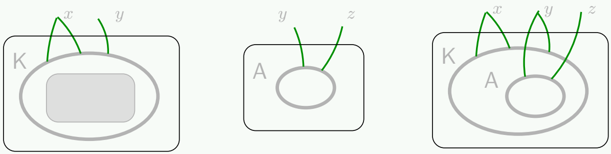
- **PRIME PRODUCT:** $G_0 \mid G_1$: like $\parallel$ with merge of roots in the resulting bigraph.

# Bigraphical term language



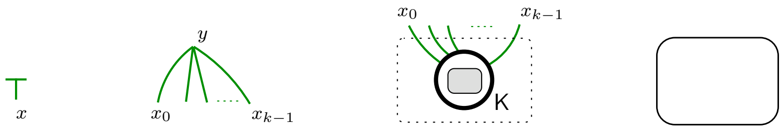

closure $/x{:}\{x\}{\to}\epsilon$ subst $y/X{:}X{\to}\{y\}$ ion $\mathsf{K}_{\vec{x}}{:}\langle 1,\emptyset\rangle{\to}\langle 1,\vec{x}\rangle$ barren root $1{:}\epsilon{\to}1$

These together with **merge** of roots and **permutations** of sites/roots provide a complete set constructors.



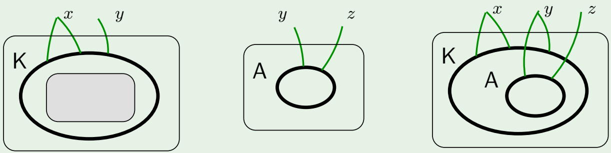

A ion $\mathsf{K}_{xxy}$, an atom $\mathsf{A}_{yz}$ and the molecule $(\mathsf{K}_{xxy} \| yz) \circ \mathsf{A}_{yz}$.

# Bigraphical term language




closure $/x\!:\!\{x\}\!\to\!\epsilon$  subst $y/X\!:\!X\!\to\!\{y\}$  ion $\mathsf{K}_{\vec{x}}\!:\!\langle 1,\emptyset\rangle\!\to\!\langle 1,\vec{x}\rangle$  barren root $1\!:\!\epsilon\!\to\!1$

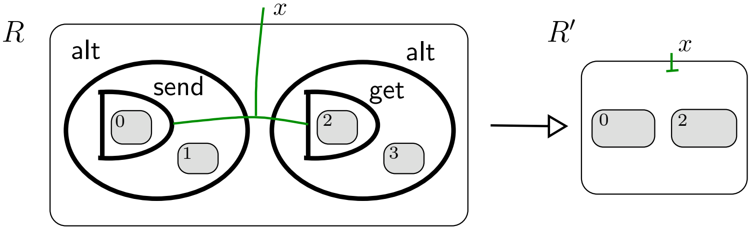
These together with **merge** of roots and **permutations** of sites/roots provide a complete set constructors.




A ion $\mathsf{K}_{xxy}$, an atom $\mathsf{A}_{yz}$ and the molecule $(\mathsf{K}_{xxy} \,\|\, yz) \circ \mathsf{A}_{yz}$.

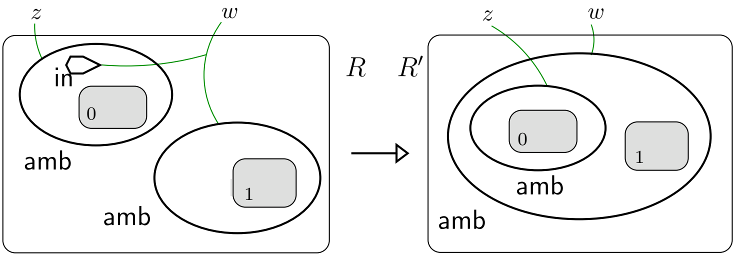
# Bigraphical reactive systems, CCS

The **dynamics** of bigraphs is not hardwired in the model, but specified each time in terms of (affine) parametric rewrite rules. Similarly to graph rewriting.



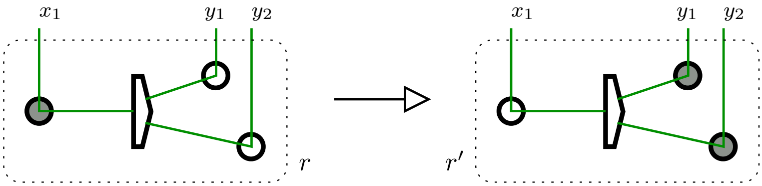

$$\mathbf{alt}(\mathbf{send}_x\square_0 \mid \square_1)|\mathbf{alt}(\mathbf{get}_x\square_2 \mid \square_3) \longrightarrow \square_0 \mid \square_2 \mid x$$

# Bigraphical reactive systems, Ambients




$$\mathsf{amb}_z(\mathsf{in}_w \mid \square_0) \mid \mathsf{amb}_w\square_1 \longrightarrow \mathsf{amb}_w(\mathsf{amb}_z\square_0 \mid \square_1)$$

# Bigraphical reactive systems, Petri nets




$$\mathbf{M}_{x_1} \mid \mathbf{E}_{x_1,y_1y_2} \mid \mathbf{U}_{y_1} \mid \mathbf{U}_{y_2} \longrightarrow \mathbf{U}_{x_1} \mid \mathbf{E}_{x_1,y_1y_2} \mid \mathbf{M}_{y_1} \mid \mathbf{M}_{y_2}$$

# Bigraphical reactive systems, formally

A **parametric reaction rule** has a redex $R$ and a reactum $R'$, and takes the following form

$$(R\colon I \to J, R'\colon I' \to J, \varrho) \ ,$$

where $\varrho$ maps sites of $R'$ back to $R$ injectively.

For $d$ a tuple of parameters, this results in a ground reaction rule

$$((\mathrm{id}_X \otimes R) \circ d, (\mathrm{id}_X \otimes R') \circ \varrho(d))$$

**Disclaimer (BASIC BIGRAPHICAL REACTIVE SYSTEMS)**

They enforce important simplifying properties of redexes: flatness (no nesting of nodes), guardedness (no inner name is open, no site has a root as parent), simpleness (no inner names are peer, no sites are siblings), and definiteness (no redex involve only a subset of the controls involved in another).

# Bigraphical reactive systems, formally

A **parametric reaction rule** has a redex $R$ and a reactum $R'$, and takes the following form

$$(R\colon I \to J, R'\colon I' \to J, \varrho) \ ,$$

where $\varrho$ maps sites of $R'$ back to $R$ injectively.

For $d$ a tuple of parameters, this results in a ground reaction rule

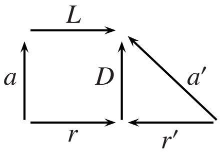
$$((\mathsf{id}_X \otimes R) \circ d, (\mathsf{id}_X \otimes R') \circ \varrho(d))$$

**Disclaimer (BASIC BIGRAPHICAL REACTIVE SYSTEMS)**

They enforce important simplifying properties of redexes: flatness (no nesting of nodes), guardedness (no inner name is open, no site has a root as parent), simpleness (no inner names are peer, no sites are siblings), and definiteness (no redex involve only a subset of the controls involved in another).

# The essence of bigraphical reactive systems

For a ground prime term $a$ and a ground reaction rule $(r, r')$, we derive a standard transition $a \xrightarrow{L} a'$ as below, where $L$ and $D$ are an idem pushout of $a$ and $r$ and $D$ is an active context.




We write $\sim_{ST}$ for the bisimilarity of the standard transition system.

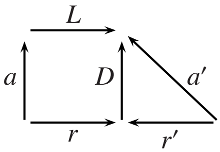
We then focus on engaged transitions, where the agent shares at least one node with the parametric redex $R$ underlying $r$. Let $\sim_{FPE}$ be the associated bisimilarity.

**Theorem**

*In any basic BRSs $\sim_{FPE}$ coincides with $\sim_{ST}$ and is a congruence.*

# The essence of bigraphical reactive systems

For a ground prime term $a$ and a ground reaction rule $(r, r')$, we derive a standard transition $a \xrightarrow{L} a'$ as below, where $L$ and $D$ are an idem pushout of $a$ and $r$ and $D$ is an active context.




We write $\sim_{\mathrm{ST}}$ for the bisimilarity of the standard transition system.

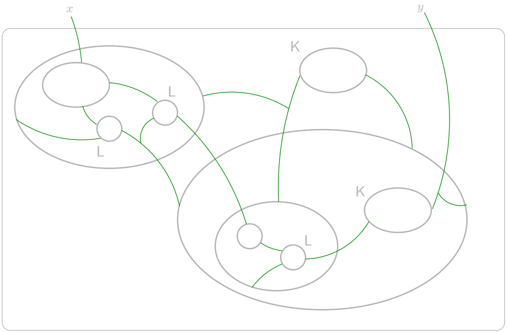
We then focus on engaged transitions, where the agent shares at least one node with the parametric redex $R$ underlying $r$. Let $\sim_{\mathrm{FPE}}$ be the associated bisimilarity.

**Theorem**

*In any basic BRSs $\sim_{\mathrm{FPE}}$ coincides with $\sim_{\mathrm{ST}}$ and is a congruence.*

# Binding bigraphs

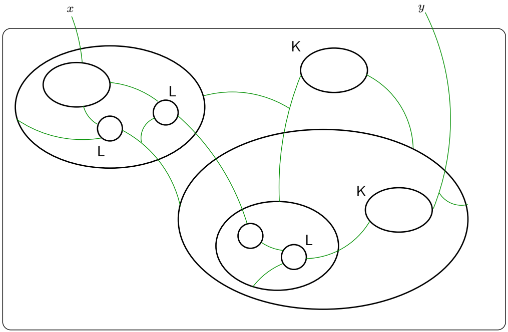
An important element is still missing in the model:
the possibility of making a name **local**.




Observe the difference between closure $/x$, where an edge 'solidifies' once and for all, and the concept that an edge is available only within certain confines. Reminiscent of communication in ambient calculus.

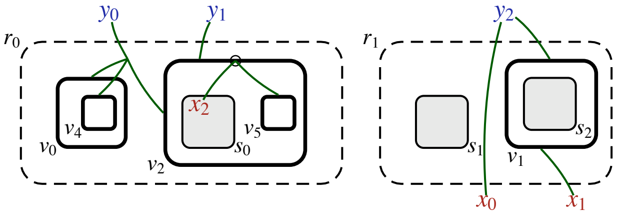
# Binding bigraphs

An important element is still missing in the model:
the possibility of making a name **local**.




Observe the difference between closure $/x$, where an edge 'solidifies' once and for all, and the concept that an edge is available only within certain confines. Reminiscent of communication in ambient calculus.

# A binding bigraph




Controls have binding ports, roots/sites have local names

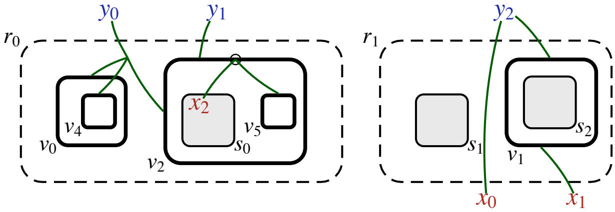
Clearly, the trick is to preserve confinement under composition.

**SCOPE RULE:** If $p$ is a binder located at a node or at root $w$, then every peer in the same link of $p$ must be contained in $w$.

New interfaces complete the picture:

$$\langle 3, (\{x_2\}, \emptyset, \emptyset), \{x_0, x_1, x_2\}\rangle \to \langle 2, (\emptyset, \emptyset), \{y_0, y_1, y_2\}\rangle$$

# A binding bigraph




Controls have binding ports, roots/sites have local names

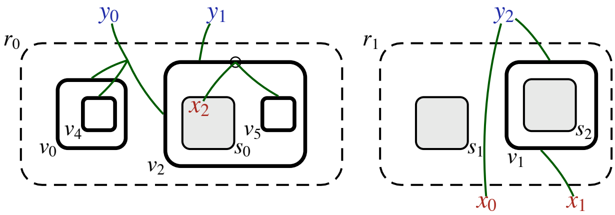
Clearly, the trick is to preserve confinement under composition.

**Scope Rule:** If $p$ is a binder located at a node or at root $w$, then every peer in the same link of $p$ must be contained in $w$.

New interfaces complete the picture:

$$\langle 3, (\{x_2\}, \emptyset, \emptyset), \{x_0, x_1, x_2\}\rangle \to \langle 2, (\emptyset, \emptyset), \{y_0, y_1, y_2\}\rangle$$

# A binding bigraph




Controls have binding ports, roots/sites have local names

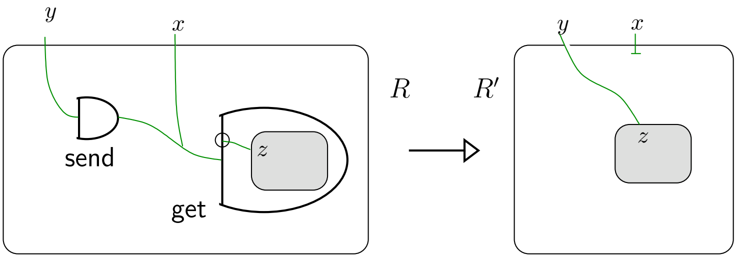
Clearly, the trick is to preserve confinement under composition.

**SCOPE RULE:** If $p$ is a binder located at a node or at root $w$, then every peer in the same link of $p$ must be contained in $w$.

New interfaces complete the picture:

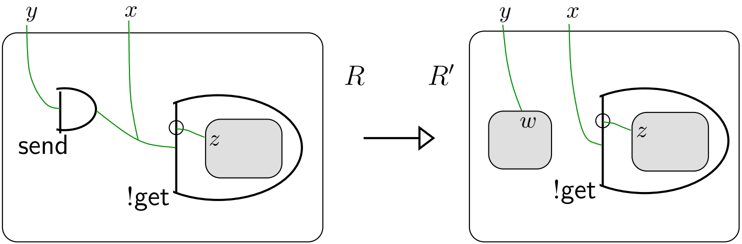
$$\langle 3, (\{x_2\}, \emptyset, \emptyset), \{x_0, x_1, x_2\}\rangle \to \langle 2, (\emptyset, \emptyset), \{y_0, y_1, y_2\}\rangle$$

# Bigraphical reactive systems, $\pi_A$



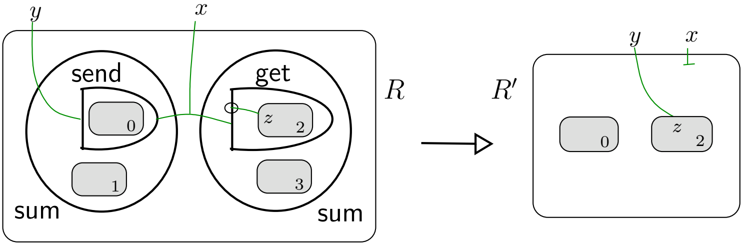

$$\mathsf{send}_{xy} \mid \mathsf{get}_{x(z)} \square \longrightarrow x \mid {}^{y}/_{(z)} \square$$

# Bigraphical reactive systems, $!\pi_A$



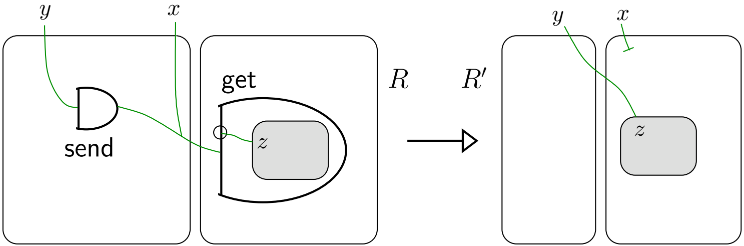

$$\mathsf{send}_{xy} \mid\ !\mathsf{get}_{x(z)}\ \square \longrightarrow {}^{y}/_{(w)}\ \square \mid\ !\mathsf{get}_{x(z)}\ \square$$

# Bigraphical reactive systems, $\pi$




$$\mathsf{sum}(\mathsf{send}_{xy}\,\square_0 \mid \square_1) \mid \mathsf{sum}(\mathsf{get}_{x(z)}\,\square_2 \mid \square_3) \longrightarrow x \mid \square_0 \mid {}^{y}/_{(z)}\,\square_2$$

# Bigraphical reactive systems, Dpi




$$\mathsf{send}_{xy} \parallel \mathsf{get}_{x(z)} \square \longrightarrow 1 \parallel x \parallel {}^{y}/_{(z)} \square$$

# Binding bigraphs, formally

**Definition** (SIGNATURE OF BINDING CONTROLS $\mathcal{K}$)

$K \in \mathcal{K}$ has **binding arity** $h$ and a **free arity** $k$, $K : h \to k$, its number of binding and non-binding ports. If $K$ is *atomic*, then $h = 0$.

**Definition** (BINDING INTERFACE)

$I = \langle m, loc, X \rangle$, where $I^{u} = \langle m, X \rangle$ is a pure interface and $loc : X \rightharpoonup m$ is a partial **locality map** which associates names in $X$ with roots. If $loc(x) = \perp$ then $x$ is **global**.

**Definition** (BINDING BIGRAPHS)

$G : I \to J$ consists of an underlying pure bigraph $G^{u} : I^{u} \to J^{u}$ which satisfies the **scope rule**, where the **binders** of $G$ are the binding ports of its nodes and the local names of its outer face $J$.

# Binding bigraphs, formally

**Definition** (SIGNATURE OF BINDING CONTROLS $\mathcal{K}$)

$K \in \mathcal{K}$ has **binding arity** $h$ and a **free arity** $k$, $K : h \to k$, its number of binding and non-binding ports. If $K$ is *atomic*, then $h = 0$.

**Definition** (BINDING INTERFACE)

$I = \langle m, loc, X \rangle$, where $I^{u} = \langle m, X \rangle$ is a pure interface and $loc : X \rightharpoonup m$ is a partial **locality map** which associates names in $X$ with roots. If $loc(x) = \bot$ then $x$ is **global**.

**Definition** (BINDING BIGRAPHS)

$G : I \to J$ consists of an underlying pure bigraph $G^{u} : I^{u} \to J^{u}$ which satisfies the **scope rule**, where the **binders** of $G$ are the binding ports of its nodes and the local names of its outer face $J$.

# Binding bigraphs, formally

**Definition (Signature of binding controls $\mathcal{K}$)**

$K \in \mathcal{K}$ has **binding arity** $h$ and a **free arity** $k$, $K\colon h \to k$, its number of binding and non-binding ports. If $K$ is *atomic*, then $h = 0$.

**Definition (Binding interface)**

$I = \langle m, loc, X \rangle$, where $I^{\mathsf{u}} = \langle m, X \rangle$ is a pure interface and $loc\colon X \rightharpoonup m$ is a partial **locality map** which associates names in $X$ with roots. If $loc(x) = \bot$ then $x$ is **global**.

**Definition (Binding bigraphs)**

$G\colon I \to J$ consists of an underlying pure bigraph $G^{\mathsf{u}}\colon I^{\mathsf{u}} \to J^{\mathsf{u}}$ which satisfies the **scope rule**, where the **binders** of $G$ are the binding ports of its nodes and the local names of its outer face $J$.

# Categorically speaking

Binding bigraphs are the arrows of a **s-category** $\acute{}\mathrm{B}{\small BG}(\mathcal{K})$, with exactly the same issues discussed for pure bigraphs.

$\mathrm{B}{\small BG}(\mathcal{K})$ is the category of **abstract binding bigraphs**, that is symmetric partial monoidal and admit a well-behaved quotient functor $[\![\cdot]\!] : \acute{}\mathrm{B}{\small BG}(\mathcal{K}) \to \mathrm{B}{\small BG}(\mathcal{K})$.

The semantic framework provided by **basic bigraphical reactive systems**, and related results, carry over to binding bigraphs, mutatis mutandis.

# Categorically speaking

Binding bigraphs are the arrows of a **s-category** $\acute{\mathrm{B}}\mathrm{BG}(\mathcal{K})$, with exactly the same issues discussed for pure bigraphs.

$\mathrm{BBG}(\mathcal{K})$ is the category of **abstract binding bigraphs**, that is symmetric partial monoidal and admit a well-behaved quotient functor $[\![\cdot]\!] : \acute{\mathrm{B}}\mathrm{BG}(\mathcal{K}) \to \mathrm{BBG}(\mathcal{K})$.

The semantic framework provided by **basic bigraphical reactive systems**, and related results, carry over to binding bigraphs, mutatis mutandis.

# Categorically speaking

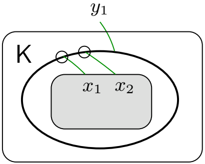
Binding bigraphs are the arrows of a **s-category** $\acute{\mathrm{B}}\mathrm{BG}(\mathcal{K})$, with exactly the same issues discussed for pure bigraphs.

$\mathrm{BBG}(\mathcal{K})$ is the category of **abstract binding bigraphs**, that is symmetric partial monoidal and admit a well-behaved quotient functor $[\![\cdot]\!] : \acute{\mathrm{B}}\mathrm{BG}(\mathcal{K}) \to \mathrm{BBG}(\mathcal{K})$.

The semantic framework provided by **basic bigraphical reactive systems**, and related results, carry over to binding bigraphs, mutatis mutandis.

# Bigraphical term language (1)

We only need to refine the notion of **ion**, and define new **wirings** to handle local names.




The ion $K_{y_1(x_1 x_2)}$ generated from a control $K : 2 \to 1$

Similarly, for each suitable choice of distinct names $\vec{x}$ and $\vec{y}$, each non-atomic control $K : h \to k$ defines a **free discrete ion**

$$K_{\vec{y}(\vec{x})} : \langle 1, (X), X \rangle \to \langle 1, (\emptyset), Y \rangle .$$

# Bigraphical term language (2)

A **prime** bigraph has no inner global names and only one root.

**Definition (CONCRETION)**

$\ulcorner X \urcorner \colon \langle 1, (X \uplus Y), X \uplus Y \rangle \to \langle 1, (Y), X \uplus Y \rangle$, a prime bigraph that globalises a subset if its local inner names.

**Definition (ABSTRACTION)**

$(X)P \colon I \to \langle 1, (X \uplus Y), Z \rangle$ localises a subset of names $X \subseteq Z$ for $P \colon I \to \langle 1, (Y), Z \rangle$ a prime.

$$/(X) = ((/X \otimes \mathrm{id}) \circ \ulcorner X \urcorner \qquad \text{closure}$$

$$\vec{y}/(\vec{x}) = (\vec{y}/\vec{x} \otimes \mathrm{id}) \circ \ulcorner \vec{x} \urcorner \qquad \text{substitution}$$

$$(\vec{y})/(\vec{x}) = (\vec{y})\vec{y}/(\vec{x}) \qquad \text{local substitution}$$

# Bigraphical term language (2)

A **prime** bigraph has no inner global names and only one root.

## Definition (CONCRETION)

$\ulcorner X \urcorner \colon \langle 1, (X \uplus Y), X \uplus Y \rangle \to \langle 1, (Y), X \uplus Y \rangle$, a prime bigraph that globalises a subset if its local inner names.

## Definition (ABSTRACTION)

$(X)P \colon I \to \langle 1, (X \uplus Y), Z \rangle$ localises a subset of names $X \subseteq Z$ for $P \colon I \to \langle 1, (Y), Z \rangle$ a prime.

$$/(X) = ((/X \otimes \mathrm{id}) \circ \ulcorner X \urcorner$$ closure

$$\vec{y}/(\vec{x}) = (\vec{y}/\vec{x} \otimes \mathrm{id}) \circ \ulcorner \vec{x} \urcorner$$ substitution

$$(\vec{y})/(\vec{x}) = (\vec{y})\vec{y}/(\vec{x})$$ local substitution

# Bigraphical term language (2)

A **prime** bigraph has no inner global names and only one root.

**Definition (CONCRETION)**

$\ulcorner X \urcorner : \langle 1, (X \uplus Y), X \uplus Y \rangle \to \langle 1, (Y), X \uplus Y \rangle$, a prime bigraph that globalises a subset if its local inner names.

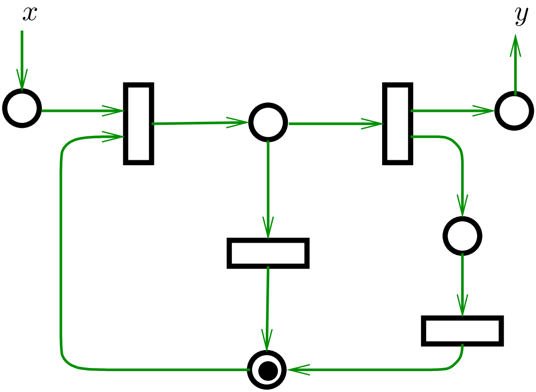
**Definition (ABSTRACTION)**

$(X)P : I \to \langle 1, (X \uplus Y), Z \rangle$ localises a subset of names $X \subseteq Z$ for $P : I \to \langle 1, (Y), Z \rangle$ a prime.

| | |
|---|---|
| $/(X) = ((/X \otimes \mathrm{id}) \circ \ulcorner X \urcorner$ | closure |
| $\vec{y}/(\vec{x}) = (\vec{y}/\vec{x} \otimes \mathrm{id}) \circ \ulcorner \vec{x} \urcorner$ | substitution |
| $(\vec{y})/(\vec{x}) = (\vec{y})\vec{y}/(\vec{x})$ | local substitution |

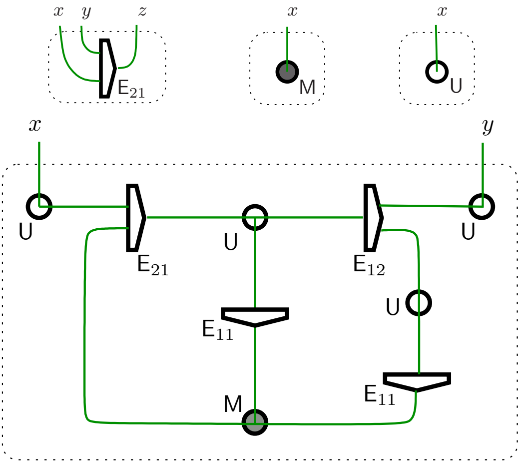
# The need for sorting

Petri nets provide an obvious illustration:



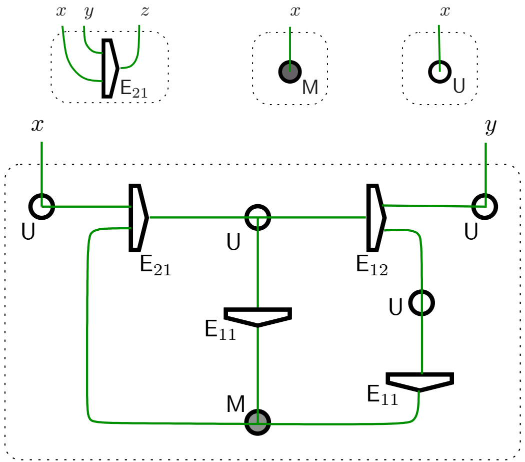

They are bipartite graphs, ports cannot be linked arbitrarily!

# Controls for Petri nets




We need to distinguish ports of different sorts and constrain the way the can be connected. We will call this a **sorting discipline**.

# Controls for Petri nets




We need to distinguish ports of different sorts and constrain the way the can be connected. We will call this a **sorting discipline**.

# Polyadic pi calculus

When modelling calculi like $\pi$, where links carry datatypes, it is equally fundamental to deploy sorting disciplines; in general, the development of a theory of link-types for bigraphs could prove rewarding.

**Polyadic pi** – The idea:

$$x(y_1, \ldots, y_n).P \mid \bar{x}\langle z_1, \ldots, z_n\rangle.Q \to \{\vec{z}/\vec{y}\}P \mid Q$$

It opened the research on typing for process calculi:

$$\overline{a}\langle b, c\rangle.P \mid a(x).Q$$
$$\overline{a}\langle \mathrm{true}\rangle.P \mid a(x).x(y).Q$$

Both terms are ill-formed, make no sense, and must therefore be ruled out. This is one of the role of types.

# Polyadic pi calculus

When modelling calculi like $\pi$, where links carry datatypes, it is equally fundamental to deploy sorting disciplines; in general, the development of a theory of link-types for bigraphs could prove rewarding.

**Polyadic pi** – The idea:

$$x(y_1, \ldots, y_n).P \mid \bar{x}\langle z_1, \ldots, z_n\rangle.Q \to \{\vec{z}/\vec{y}\}P \mid Q$$

It opened the research on typing for process calculi:

$$\overline{a}\langle b, c\rangle.P \mid a(x).Q$$
$$\overline{a}\langle \mathrm{true}\rangle.P \mid a(x).x(y).Q$$

Both terms are ill-formed, make no sense, and must therefore be ruled out. This is one of the role of types.

# Milner's sorting system

## A sorting system

- A function $\Sigma : S \longrightarrow S^*$ describes the tuples allowed on channels of each sort. $\Sigma(\gamma)$ is the object sort of $\gamma$.
- Object sort of $\gamma \in S$ must follow the sorting discipline $\Sigma(\gamma)$.
- $P$ respects $\Sigma$ if in each subterm $\overline{x}\langle\vec{y}\rangle.P'$ or $x(\vec{y}).P'$, if $x : \gamma$, then $\vec{y} : \Sigma(\gamma)$

**Subject Reduction:**

If $P$ respects $\Sigma$ and $P \longrightarrow Q$, then $Q$ respects $\Sigma$.

It follows that $P \xrightarrow{\overline{x}\vec{y}}$ implies that $\vec{y} : \Sigma(\gamma)$, for $x : \gamma$.

Therefore, these cannot happen:

$$\overline{a}\langle b, c\rangle.P \mid a(x).Q$$
$$\overline{a}\langle \mathrm{true}\rangle.P \mid a(x).x(y).Q$$

# Milner's sorting system

## A sorting system

- A function $\Sigma : S \longrightarrow S^*$ describes the tuples allowed on channels of each sort. $\Sigma(\gamma)$ is the object sort of $\gamma$.
- Object sort of $\gamma \in S$ must follow the sorting discipline $\Sigma(\gamma)$.
- $P$ respects $\Sigma$ if in each subterm $\overline{x}\langle\vec{y}\rangle.P'$ or $x(\vec{y}).P'$, if $x : \gamma$, then $\vec{y} : \Sigma(\gamma)$

## Subject Reduction:

If $P$ respects $\Sigma$ and $P \longrightarrow Q$, then $Q$ respects $\Sigma$.

It follows that $P \xrightarrow{\overline{x}\vec{y}}$ implies that $\vec{y} : \Sigma(\gamma)$, for $x : \gamma$.

Therefore, these cannot happen:

$$\overline{a}\langle b, c\rangle.P \mid a(x).Q$$
$$\overline{a}\langle \mathrm{true}\rangle.P \mid a(x).x(y).Q$$

# Simply typed pi calculus

$$S, T ::= \mathrm{B} \quad \text{types of basic values}$$
$$\mid (T_1, \ldots, T_k) \quad \text{tuple type, } k \geqslant 0$$
$$\mid \sharp T \quad \text{link type (channel)}$$

## Channel types

- inform on the type of the value they carry

## Examples

- $\sharp(\mathrm{int})$ : channel carrying values of type int.
- $\sharp(\mathrm{unit})$ : channel carrying $\star$, the only value of type unit.
- $\sharp(\sharp\mathrm{int})$ : channel carrying channels carrying integers.

# IO types and subtypes

$$\begin{array}{rcll} S, T & ::= & \ldots & \\ & \mid & \mathrm{r}T & \text{input capability on a channel of } T \text{ values} \\ & \mid & \mathrm{w}T & \text{output capability on a channel of } T \text{ values} \\ & \mid & \sharp T & \text{link type (channel)} \end{array}$$

Channel types:

- inform on the type of the value they carry
- offer capabilities to their users

Examples:

- r(int) : input-only channel carrying values of type int.
- $\sharp$r(int) : channel carrying input-only integer channels.

Subtyping kicks in: any channel can be used in only one of its capabilities...

$$\sharp T \leq \mathrm{r}T, \mathrm{w}T$$

Subsumption: if $x : \sharp T$, then $x : \mathrm{r}T$ and $x : \mathrm{w}T$ too

# IO types and subtypes

$$
\begin{array}{rcll}
S, T & ::= & \ldots & \\
& | & \mathsf{r}T & \text{input capability on a channel of } T \text{ values} \\
& | & \mathsf{w}T & \text{output capability on a channel of } T \text{ values} \\
& | & \sharp T & \text{link type (channel)}
\end{array}
$$

Channel types:

- inform on the type of the value they carry
- offer capabilities to their users

Examples:

- r(int) : input-only channel carrying values of type int.
- ♯r(int) : channel carrying input-only integer channels.

Subtyping kicks in: any channel can be used in only one of its capabilities. . .

$$\sharp T \leq \mathsf{r}T, \mathsf{w}T$$

Subsumption: if $x : \sharp T$, then $x : \mathsf{r}T$ and $x : \mathsf{w}T$ too

# IO types for polyadic pi

We now develop those ideas (in a slightly different syntax, in particular no basic types).

$$\frac{}{()^{I} :: Type} \qquad \frac{T_1 \ldots T_n :: Type}{(T_1, \ldots, T_n)^{\mathrm{r}} :: Type} \qquad \frac{T_1 \ldots T_n :: Type}{(T_1, \ldots, T_n)^{\mathrm{w}} :: Type}$$

$$\frac{T_1 \ldots T_n :: Type \qquad S_1 \ldots S_n :: Type \qquad S_i \leq T_i}{(T_1, \ldots, T_n; S_1, \ldots, S_n)^{\mathrm{b}} :: Type}$$

We will refer to these types as the set of sorts $\mathcal{S}$.

# The subtyping relation

$$\frac{T_i \le T'_i, \quad i = 1, \dots, n}{(T_1, \dots, T_n)^{\mathrm{r}} \le (T'_1, \dots, T'_n)^{\mathrm{r}}} \qquad \frac{T_i \le T'_i, \quad i = 1, \dots, n}{(T'_1, \dots, T'_n)^{\mathrm{w}} \le (T_1, \dots, T_n)^{\mathrm{w}}}$$

$$\frac{T_i \le T'_i \ \text{ and } \ S_i \le S'_i, \quad i = 1, \dots, n}{(T_1, \dots, T_n; S'_1, \dots, S'_n)^{\mathrm{b}} \le (T'_1, \dots, T'_n; S_1, \dots, S_n)^{\mathrm{b}}}$$

$$\frac{T_i \le T'_i, \quad i = 1, \dots, n}{(T_1, \dots, T_n; S_1, \dots, S_n)^{\mathrm{b}} \le (T'_1, \dots, T'_n)^{\mathrm{r}}}$$

$$\frac{T_i \le T'_i, \quad i = 1, \dots, n}{(S_1, \dots, S_n; T'_1, \dots, T'_n)^{\mathrm{b}} \le (T_1, \dots, T_n)^{\mathrm{w}}}$$

# The typing judgement

$$\frac{}{\Gamma \vdash \mathbf{0} : \circ} \qquad \frac{\Gamma \vdash P : \circ \quad \Gamma \vdash Q : \circ}{\Gamma \vdash P \mid Q : \circ} \qquad \frac{\Gamma, n \colon S \vdash P : \circ}{\Gamma \vdash (\nu n \colon S)P : \circ}$$

$$\frac{\Gamma(n) \leq (S_1, \ldots, S_n)^{\mathrm{r}} \qquad \Gamma, m_1 : S_1, \ldots, m_n \colon S_n \vdash P : \circ}{\Gamma \vdash n(m_1 : S_1, \ldots, m_n \colon S_n).P : \circ}$$

$$\frac{\Gamma(n) \leq (\Gamma(m_1), \ldots, \Gamma(m_n))^{\mathrm{w}} \qquad \Gamma \vdash P : \circ}{\Gamma \vdash \overline{n}\langle m_1, \ldots, m_n \rangle.P : \circ}$$

**Lemma** (SUBJECT REDUCTION FOR POLYADIC PI)

*If* $\Gamma \vdash P : \circ$, *then* $\Gamma \vdash P' : \circ$ *for each* $P \rightarrow_{\pi} P'$.

# Sorting edges

Extend bigraphs with a sorting structure suitable to represent the polyadic pi calculus; then describe such a representation.

(FIRST STEP: EDGE SIGNATURES)

An edge signature $\mathcal{E}$ is a set of so-called **edge controls**. Edges are now assigned controls the same way nodes are.

All previous development carries over to the s-category $\acute{\mathrm{B}}\mathrm{BG}(\mathcal{K}, \mathcal{E})$ and the category $\mathrm{BBG}(\mathcal{K}, \mathcal{E})$.

(SECOND STEP: SORTED SIGNATURES)

Let $\Theta$ denote a non-empty set of **sorts**; each $E \in \mathcal{E}$ is ascribed a sort $\theta \in \Theta$; we say that $\mathcal{E}$ is $\Theta$-sorted.

# Sorting edges

Extend bigraphs with a sorting structure suitable to represent the polyadic pi calculus; then describe such a representation.

(FIRST STEP: EDGE SIGNATURES)

An edge signature $\mathcal{E}$ is a set of so-called **edge controls**. Edges are now assigned controls the same way nodes are.

All previous development carries over to the s-category $\acute{\text{B}}\text{BG}(\mathcal{K}, \mathcal{E})$ and the category $\text{BBG}(\mathcal{K}, \mathcal{E})$.

(SECOND STEP: SORTED SIGNATURES)

Let $\Theta$ denote a non-empty set of **sorts**; each $E \in \mathcal{E}$ is ascribed a sort $\theta \in \Theta$; we say that $\mathcal{E}$ is $\Theta$-sorted.

# Sorted binding bigraphs

**Definition (Θ-SORTED BINDING BIGRAPHS)**

A binding bigraph in ´BBG$(\mathcal{K}, \mathcal{E})$ with both interfaces enriched by ascribing a $\Theta$-sort to each name. Observe that ports inherit their sorting from the link they are connected to.

´SBBG$(\Theta, \mathcal{K}, \mathcal{E})$ is the s-category of sorted binding bigraphs.

**Definition (SORTING DISCIPLINE)**

$\Sigma = (\Theta, \mathcal{K}, \mathcal{E}, \Phi)$, where $\Phi$ is a 'well-formedness' condition on $\Theta$-sorted binding bigraphs that is satisfied by the identities and preserved by both composition and tensor product.

A binding bigraph is **$\Sigma$-sorted** if it satisfies $\Phi$.

$\Sigma$-sorted bigraphs form a sub-s-category of ´SBBG$(\Theta, \mathcal{K}, \mathcal{E})$ denoted by ´SBBG$(\Sigma)$. Importantly, the results on bisimulation must be extended to $\Sigma$-sorted BRSs to ´SBBG$(\Sigma)$.

# Sorted binding bigraphs

**Definition (Θ-SORTED BINDING BIGRAPHS)**

A binding bigraph in ´$\mathrm{BBG}(\mathcal{K}, \mathcal{E})$ with both interfaces enriched by ascribing a $\Theta$-sort to each name. Observe that ports inherit their sorting from the link they are connected to.

´$\mathrm{SBBG}(\Theta, \mathcal{K}, \mathcal{E})$ is the s-category of sorted binding bigraphs.

**Definition (SORTING DISCIPLINE)**

$\Sigma = (\Theta, \mathcal{K}, \mathcal{E}, \Phi)$, where $\Phi$ is a 'well-formedness' condition on $\Theta$-sorted binding bigraphs that is satisfied by the identities and preserved by both composition and tensor product.

A binding bigraph is **$\Sigma$-sorted** if it satisfies $\Phi$.

$\Sigma$-sorted bigraphs form a sub-s-category of ´$\mathrm{SBBG}(\Theta, \mathcal{K}, \mathcal{E})$ denoted by ´$\mathrm{SBBG}(\Sigma)$. Importantly, the results on bisimulation must be extended to $\Sigma$-sorted BRSs to ´$\mathrm{SBBG}(\Sigma)$.

# Sorted binding bigraphs

**Definition (Θ-SORTED BINDING BIGRAPHS)**

A binding bigraph in $\acute{\mathrm{B}}\mathrm{BG}(\mathcal{K}, \mathcal{E})$ with both interfaces enriched by ascribing a $\Theta$-sort to each name. Observe that ports inherit their sorting from the link they are connected to.

$\acute{\mathrm{S}}\mathrm{BBG}(\Theta, \mathcal{K}, \mathcal{E})$ is the s-category of sorted binding bigraphs.

**Definition (SORTING DISCIPLINE)**

$\Sigma = (\Theta, \mathcal{K}, \mathcal{E}, \Phi)$, where $\Phi$ is a 'well-formedness' condition on $\Theta$-sorted binding bigraphs that is satisfied by the identities and preserved by both composition and tensor product.

A binding bigraph is **$\Sigma$-sorted** if it satisfies $\Phi$.

$\Sigma$-sorted bigraphs form a sub-s-category of $\acute{\mathrm{S}}\mathrm{BBG}(\Theta, \mathcal{K}, \mathcal{E})$ denoted by $\acute{\mathrm{S}}\mathrm{BBG}(\Sigma)$. Importantly, the results on bisimulation must be extended to $\Sigma$-sorted BRSs to $\acute{\mathrm{S}}\mathrm{BBG}(\Sigma)$.

# A sorting discipline for polyadic pi

**Definition** (SUBSORTING $(\Theta, \mathcal{K}, \mathcal{E}, \Phi)$)

Let $\Theta$ be a preorder of sorts, and $Q$ a set of type constructors.

Each $K$ is associated with $q_K \in Q$ and a partition of its ports into two sets, $\mathcal{C}_K$ and $\mathcal{V}_K$: of **communication** and **value** ports.

If $q$ is covariant on $i$ and $K$'s $i$th port is a value port, then it must be a **binding** port.

Let $\mathcal{E}$ deliver an arbitrary assignment of sorts to edge controls, and condition $\Phi$ be as follows:

- for each inner name $x : S$, if $T$ is the sort of its link, then $T \leq S$.
- for each $K$-node $v$ and each $c \in \mathcal{C}_K$, let $S$ be the sort of $v$'s corresponding port; we require that $S \leq q_K(S_1, \ldots, S_{|\mathcal{V}_K|})$, where $S_i$ is the sort of $v$'s $i$th port in $\mathcal{V}_K$.

These data determine a suitable sorting.

# A sorting discipline for polyadic pi

**Definition (SUBSORTING $(\Theta, \mathcal{K}, \mathcal{E}, \Phi)$)**

Let $\Theta$ be a preorder of sorts, and $Q$ a set of type constructors.

Each $K$ is associated with $q_K \in Q$ and a partition of its ports into two sets, $\mathcal{C}_K$ and $\mathcal{V}_K$: of **communication** and **value** ports.

If $q$ is covariant on $i$ and $K$'s $i$th port is a value port, then it must be a **binding** port.

Let $\mathcal{E}$ deliver an arbitrary assignment of sorts to edge controls, and condition $\Phi$ be as follows:

- for each inner name $x : S$, if $T$ is the sort of its link, then $T \leq S$.
- for each $K$-node $v$ and each $c \in \mathcal{C}_K$, let $S$ be the sort of $v$'s corresponding port; we require that $S \leq q_K(S_1, \ldots, S_{|\mathcal{V}_K|})$, where $S_i$ is the sort of $v$'s $i$th port in $\mathcal{V}_K$.

These data determine a suitable sorting.

# A sorting discipline for polyadic pi

**Definition (SUBSORTING $(\Theta, \mathcal{K}, \mathcal{E}, \Phi)$)**

Let $\Theta$ be a preorder of sorts, and $Q$ a set of type constructors.

Each $K$ is associated with $q_K \in Q$ and a partition of its ports into two sets, $\mathcal{C}_K$ and $\mathcal{V}_K$: of **communication** and **value** ports.

If $q$ is covariant on $i$ and $K$'s $i$th port is a value port, then it must be a **binding** port.

Let $\mathcal{E}$ deliver an arbitrary assignment of sorts to edge controls, and condition $\Phi$ be as follows:

- for each inner name $x \colon S$, if $T$ is the sort of its link, then $T \leq S$.
- for each $K$-node $v$ and each $c \in \mathcal{C}_K$, let $S$ be the sort of $v$'s corresponding port; we require that $S \leq q_K(S_1, \ldots, S_{|\mathcal{V}_K|})$, where $S_i$ is the sort of $v$'s $i$th port in $\mathcal{V}_K$.

These data determine a suitable sorting.

# A sorting discipline for polyadic pi

**Definition (SUBSORTING $(\Theta, \mathcal{K}, \mathcal{E}, \Phi)$)**

Let $\Theta$ be a preorder of sorts, and $Q$ a set of type constructors.

Each $K$ is associated with $q_K \in Q$ and a partition of its ports into two sets, $\mathcal{C}_K$ and $\mathcal{V}_K$: of **communication** and **value** ports.

If $q$ is covariant on $i$ and $K$'s $i$th port is a value port, then it must be a **binding** port.

Let $\mathcal{E}$ deliver an arbitrary assignment of sorts to edge controls, and condition $\Phi$ be as follows:

- for each inner name $x : S$, if $T$ is the sort of its link, then $T \leq S$.
- for each $K$-node $v$ and each $c \in \mathcal{C}_K$, let $S$ be the sort of $v$'s corresponding port; we require that $S \leq q_K(S_1, \ldots, S_{|\mathcal{V}_K|})$, where $S_i$ is the sort of $v$'s $i$th port in $\mathcal{V}_K$.

These data determine a suitable sorting.

# A sorting discipline for polyadic pi

**Definition (SUBSORTING $(\Theta, \mathcal{K}, \mathcal{E}, \Phi)$)**

Let $\Theta$ be a preorder of sorts, and $Q$ a set of type constructors.

Each $K$ is associated with $q_K \in Q$ and a partition of its ports into two sets, $\mathcal{C}_K$ and $\mathcal{V}_K$: of **communication** and **value** ports.

If $q$ is covariant on $i$ and $K$'s $i$th port is a value port, then it must be a **binding** port.

Let $\mathcal{E}$ deliver an arbitrary assignment of sorts to edge controls, and condition $\Phi$ be as follows:

- for each inner name $x\colon S$, if $T$ is the sort of its link, then $T \leq S$.
- for each $K$-node $v$ and each $c \in \mathcal{C}_K$, let $S$ be the sort of $v$'s corresponding port; we require that $S \leq q_K(S_1, \ldots, S_{|\mathcal{V}_K|})$, where $S_i$ is the sort of $v$'s $i$th port in $\mathcal{V}_K$.

These data determine a suitable sorting.

# A sorting discipline for polyadic pi

**Definition (SUBSORTING $(\Theta, \mathcal{K}, \mathcal{E}, \Phi)$)**

Let $\Theta$ be a preorder of sorts, and $Q$ a set of type constructors.

Each $K$ is associated with $q_K \in Q$ and a partition of its ports into two sets, $\mathcal{C}_K$ and $\mathcal{V}_K$: of **communication** and **value** ports.

If $q$ is covariant on $i$ and $K$'s $i$th port is a value port, then it must be a **binding** port.

Let $\mathcal{E}$ deliver an arbitrary assignment of sorts to edge controls, and condition $\Phi$ be as follows:

- for each inner name $x \colon S$, if $T$ is the sort of its link, then $T \leq S$.
- for each $K$-node $v$ and each $c \in \mathcal{C}_K$, let $S$ be the sort of $v$'s corresponding port; we require that $S \leq q_K(S_1, \ldots, S_{|\mathcal{V}_K|})$, where $S_i$ is the sort of $v$'s $i$th port in $\mathcal{V}_K$.

These data determine a suitable sorting.

# Bigraphical representation of polyadic pi

### Definition ($\Sigma_{\pi_\leq}$: INSTANTIATING SUBSORTING)

Let our chosen set of sorts be $\mathcal{S}$, with $Q$ the set of tags $\{b, r, w\}$ and

$$q(S_0, \ldots, S_n) = (S_0, \ldots, S_n)^q \ .$$

The signature $\mathcal{E}_{\pi_\leq}$ provides controls in correspondence with the sorts in $\mathcal{S}$. The signature $\mathcal{K}_{\pi_\leq}$ has countably many controls:

$$\textbf{send} : 0 \to (i+1) \qquad\qquad \textbf{get} : i \to 1 \ .$$

All ports but the first ones are value ports; **send** controls are associated with w, and their value ports are contravariant; **get** controls with r, and their value ports are covariant (and binding).

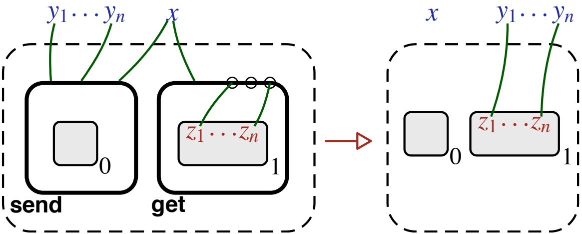
This defines ´$\mathrm{SBBG}(\Sigma_{\pi_\leq})$ (and $\mathrm{SBBG}(\Sigma_{\pi_\leq})$, of course).

# Sorted bigraphical reactive system for pi

(SORTED BRS: ´SBBG$_{\pi_\leq}$ )

$\mathcal{R}_{\pi_\leq}$ consists of a family of $\Sigma_{\pi_\leq}$-sorted reaction rules below.

The outer names $y_1, \ldots, y_n$ have sorts $T_1, \ldots, T_n$, the local names $z_1, \ldots, z_n$ and the edges they are connected to have sort $U_1, \ldots, U_n$, and the name $x$ has sort $(U_1, \ldots, U_n; T_1, \ldots, T_n)^b$.



# The encoding

$$[\![\Gamma \vdash \mathbf{0} : \circ]\!] = \mathrm{dom}(\Gamma) : \epsilon \to \langle 1, (), \Gamma \rangle$$

$$[\![\Gamma \vdash P \mid Q : \circ]\!] = [\![\Gamma \vdash P : \circ]\!] \mid [\![\Gamma \vdash Q : \circ]\!]$$

$$[\![\Gamma \vdash (\nu n\colon S)P : \circ]\!] = (/n : S)[\![\Gamma, n : S \vdash P : \circ]\!]$$

$$[\![\Gamma \vdash \overline{n}\langle \vec{m} \rangle.P : \circ]\!] = (\mathbf{send}_{n,\vec{m}} \otimes \mathrm{id}) \circ [\![\Gamma \vdash P : \circ]\!]$$

$$[\![\Gamma \vdash n(\vec{m}\colon \vec{S}).P : \circ]\!] = (\mathbf{get}_{n(\vec{m})} \otimes \mathrm{id}) \circ (\vec{m})[\![\Gamma, \vec{m}\colon \vec{S} \vdash P : \circ]\!]$$

**Proposition** (Static Correspondence)

$\Gamma \vdash P : \circ \equiv_\pi \Gamma \vdash P' : \circ$ if and only if $[\![\Gamma \vdash P : \circ]\!] = [\![\Gamma \vdash P' : \circ]\!]$.

**Theorem** (Dynamic Correspondence)

*For each well-typed* $\Gamma \vdash P : \circ$ *and agent* $a\colon \epsilon \to \langle \Gamma \rangle$, *we have*

$$[\![\Gamma \vdash P : \circ]\!] \longrightarrow\!\!\!\triangleright\; a \quad \textit{if and only if} \quad P \to_\pi a_\pi$$

# The encoding

$$[\![\Gamma \vdash \mathbf{0} : \circ]\!] = \mathrm{dom}(\Gamma) : \epsilon \to \langle 1, (), \Gamma \rangle$$

$$[\![\Gamma \vdash P \mid Q : \circ]\!] = [\![\Gamma \vdash P : \circ]\!] \mid [\![\Gamma \vdash Q : \circ]\!]$$

$$[\![\Gamma \vdash (\nu n \colon S)P : \circ]\!] = (/n : S)[\![\Gamma, n : S \vdash P : \circ]\!]$$

$$[\![\Gamma \vdash \overline{n}\langle \vec{m} \rangle.P : \circ]\!] = (\mathbf{send}_{n,\vec{m}} \otimes \mathrm{id}) \circ [\![\Gamma \vdash P : \circ]\!]$$

$$[\![\Gamma \vdash n(\vec{m} \colon \vec{S}).P : \circ]\!] = (\mathbf{get}_{n(\vec{m})} \otimes \mathrm{id}) \circ (\vec{m})[\![\Gamma, \vec{m} \colon \vec{S} \vdash P : \circ]\!]$$

**Proposition (STATIC CORRESPONDENCE)**

$\Gamma \vdash P : \circ \equiv_\pi \Gamma \vdash P' : \circ$ if and only if $[\![\Gamma \vdash P : \circ]\!] = [\![\Gamma \vdash P' : \circ]\!]$.

**Theorem (DYNAMIC CORRESPONDENCE)**

*For each well-typed* $\Gamma \vdash P : \circ$ *and agent* $a \colon \epsilon \to \langle \Gamma \rangle$, *we have*

$$[\![\Gamma \vdash P : \circ]\!] \rightarrowtriangle a \quad \textit{if and only if} \quad P \to_\pi a_\pi$$

# The encoding

$$[\![\Gamma \vdash \mathbf{0} : \circ]\!] = \mathrm{dom}(\Gamma) : \epsilon \to \langle 1, (), \Gamma \rangle$$

$$[\![\Gamma \vdash P \mid Q : \circ]\!] = [\![\Gamma \vdash P : \circ]\!] \mid [\![\Gamma \vdash Q : \circ]\!]$$

$$[\![\Gamma \vdash (\nu n\colon S)P : \circ]\!] = (/n : S)[\![\Gamma, n : S \vdash P : \circ]\!]$$

$$[\![\Gamma \vdash \overline{n}\langle \vec{m} \rangle.P : \circ]\!] = (\mathbf{send}_{n,\vec{m}} \otimes \mathrm{id}) \circ [\![\Gamma \vdash P : \circ]\!]$$

$$[\![\Gamma \vdash n(\vec{m}\colon \vec{S}).P : \circ]\!] = (\mathbf{get}_{n(\vec{m})} \otimes \mathrm{id}) \circ (\vec{m})[\![\Gamma, \vec{m}\colon \vec{S} \vdash P : \circ]\!]$$

**Proposition (STATIC CORRESPONDENCE)**

$\Gamma \vdash P : \circ \equiv_\pi \Gamma \vdash P' : \circ$ if and only if $[\![\Gamma \vdash P : \circ]\!] = [\![\Gamma \vdash P' : \circ]\!]$.

**Theorem (DYNAMIC CORRESPONDENCE)**

*For each well-typed* $\Gamma \vdash P : \circ$ *and agent* $a\colon \epsilon \to \langle \Gamma \rangle$, *we have*

$$[\![\Gamma \vdash P : \circ]\!] \longrightarrow a \quad \textit{if and only if} \quad P \to_\pi a_\pi$$

# Labels arising from engaged transitions

It follows from the general theory that $\sim_{\text{FPE}}$ is a congruence. We characterise the labels of its transition system.

**Lemma (Characterising transitions in $\text{SBBG}_{\pi_\leq}$)**

*Let $a \xrightarrow{L} a'$ be an engaged transitions. Then*

$$a = (/Z\colon \tilde{S})(r_a \mid b) \qquad\qquad L = \langle\sigma\rangle \mid r_L$$

$$a' = \sigma(/Z\colon \tilde{S})(y_1 \ldots y_n/(z_1 \ldots z_n)c_2 \mid c_1 \mid b)$$

*where, up to renaming...*

| $r_a$ | $r_L$ | $\sigma$ |
|---|---|---|
| $\mathbf{send}_{xy_1\ldots y_n} c_1$ | $\mathbf{get}_{x(z_1\ldots z_n)} c_2$ | $\mathrm{sub}_X$ |
| $\mathbf{get}_{x(z_1\ldots z_n)} c_2$ | $\mathbf{send}_{xy_1\ldots y_n} c_1$ | $\mathrm{sub}_X$ |
| $\mathbf{send}_{x_0 y_1\ldots y_n} c_1 \mid \mathbf{get}_{x_1(z_1\ldots z_n)} c_2$ | $1$ | $\mathrm{sub}_{X\setminus\{x_{\bar{i}}\}} \mid x_i/x_{\bar{i}}$ |
| $\mathbf{send}_{xy_1\ldots y_n} c_1 \mid \mathbf{get}_{x(z_1\ldots z_n)} c_2$ | $1$ | $\mathrm{sub}_X$ |

# Conclusion and further work

- We have reviewed the main ideas of **bigraphs**, with the declared intention to lure people into working on them.
- We have introduced **edge-sorting** and described subsorting, a particular sorting discipline suitable for the representation of the polyadic pi calculus with subtyping.
- We have shown such a representation and studied its properties.

- Compare $\sim_{\mathsf{FPE}}$ with standard **typed bisimulation**.
- Examine more **advanced type systems** for presentation as sortings in bigraphs; e.g., recursive and linear types.
- Consider **behavioural** types (hard).
- Identify more general **sufficient conditions** on type systems and their features that allow to present them as sortings in a bigraphical reactive system.

# Conclusion and further work

- We have reviewed the main ideas of **bigraphs**, with the declared intention to lure people into working on them.
- We have introduced **edge-sorting** and described subsorting, a particular sorting discipline suitable for the representation of the polyadic pi calculus with subtyping.
- We have shown such a representation and studied its properties.

- Compare $\sim_{\mathsf{FPE}}$ with standard **typed bisimulation**.
- Examine more **advanced type systems** for presentation as sortings in bigraphs; e.g., recursive and linear types.
- Consider **behavioural** types (hard).
- Identify more general **sufficient conditions** on type systems and their features that allow to present them as sortings in a bigraphical reactive system.

# Essential bibliography


R. Milner.
Pure bigraphs: Structure and dynamics.
*Information and Computation*, 204(1):60–122, 2006.

V. Sassone and P. Sobociński.
Locating Reactions using 2-Categories.
*Theoretical Computer Science* 333(1-2), 297–327, 2005.

O. H. Jensen and R. Milner.
Bigraphs and mobile processes (revised).
Tech Rep UCAM-CL-TR-580, University of Cambridge, Computer Lab 2004.

J. J. Leifer and R. Milner.
Transition systems, link graphs and Petri nets.
Tech Rep UCAM-CL-TR-598, University of Cambridge, Computer Lab 2004.

O. H. Jensen.
*Mobile Processes in Bigraphs.*
PhD thesis, Dept of Computer Science, Aalborg University, 2006.