Bigraphical Reactive Systems

Marino Miculan
MADS lab
(with results by many people)

MeMo Workshop, June 6, 2014

Bigraphical Reactive System

A discrete *reactive system* is composed by a set of states and a transition relation.

A ***Bigraphical Reactive System*** is a RS where

- States are **bigraphs:** data structures rendering explicitly the positions and connections of system's components
- State transitions are **bigraph rewritings** defined by a set of local reaction rules

So BRSs propose as an *operational* metamodel.

Statics

Dynamics

Metatheory & Tools

Conclusions

Bigraphical Reactive Systems

Marino Miculan
MADS lab
(with results by many people)

MeMo Workshop, June 6, 2014

Models for concurrent, distributed systems

Distributed systems are cool, but so damn complex and error prone...

In the last 40 years, TCS have developed hundreds of models and calculi, each focusing on some specific aspects, and providing

- mathematical theories
- prototypes (simulation tools)
- model checkers
- editors
- other nice stuff (even some Right Stuff!)

Models for concurrent, distributed systems

Models are cool, but still so damn complex and error prone...

- in different models, many definitions and results look almost the same
- everytime we have to start over (almost) from scratch
- Implementing tools is time consuming!

"The final model" does not exist
We have to accept a plethora of specific models

Bigraphs and Bigraphical Reactive Systems

Introduced by R. Milner et al. (2001) as a formal, graphical **meta-model** for (distributed) systems (but lot of work by many people, since then)

Main aims:

a theoretical framework covering many models dealing with *localities* and *connections*

general results, tools and techniques which can be readily instantiated to specific calculi

a common ground where different models can be formally compared and merged

Another (unexpected?) application:
a data structure for representing *semi-structured informations*, like knowledge, architectures,...

In this tutorial: a gentle (and not very abstract) introduction to bigraphs

For more details:

- Milner's book "The Space and Motion of Communicating Agent", 2009
- bigraph.org
- many other works...

a theoretical framework covering many models dealing with *localities* and *connections*

k
s

general results,
tools and techniques
which can be readily
instantiated to
specific calculi

a common ground where different models can be formally compared and merged

Another (unexpected?) application:
a data structure for representing *semi-structured informations*, like knowledge, architectures,...

In this tutorial: a gentle (and not very abstract) introduction to bigraphs

For more details:

- Milner's book "The Space and Motion of Communicating Agent", 2009
- bigraph.org
- many other works...

Bigraphical Reactive System

A discrete *reactive system* is composed by a set of states and a transition relation.

A ***Bigraphical Reactive System*** is a RS where

- States are **bigraphs:** data structures rendering explicitly the positions and connections of system's components
- State transitions are **bigraph rewritings** defined by a set of local reaction rules

So BRSs propose as an *operational* metamodel.

Statics

What's a Bigraph?

A **bigraph** G is a pair of graphs on the same (finite) set V of nodes:

the **place graph**: a forest representing the topology of the system

the **link graph**: a hypergraph representing the connections

What's a Bigraph?

Compact notation: place graph is represented by nesting nodes; sites are grey holes, and roots (or regions) are outlined

Anatomy of a Bigraph

Horizontal composition of bigraphs (tensor)

Vertical Composition of bigraphs

Bigraphs form a monoidal category

How to encode a process algebra - syntax

Basic strategy

Process Algebra	Bigraphs
syntactic constructor with n variables	control with arity n
syntactic tree	place graph
variable	outer name
name	edge

Useful shorthands: nil is just "nothing", | (parallel) can be "omitted";

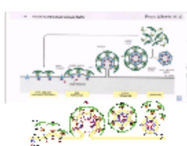
Example: CCS

Original syntax: $P ::= 0 \mid a.P \mid \bar{a}.P \mid P|Q$

Bigraphs:
Signature has two controls: *send*, *recv*

Example: encoding of

Modeling "informal" systems



This is a way to give a formal syntax to informal systems

Sortings

Often, bigraphs generated by a signature are too many.
Sorting = discipline for ruling out unwanted bigraphs

$\frac{\text{sorted bigraphs}}{\text{unsorted bigraphs}} = \frac{\text{well-formed terms}}{\text{context-free terms}}$

Can be specified in several ways (e.g. predicates in some logic); see work by Hildebrandt, Debois, Perrone, ...

Example: binding bigraphs

Some ports of controls are marked as *binding*

Requirement over bigraphs:
"all points linked to a binding port of a node *u* lie inside *u*"

Example: encoding of π-calculus terms

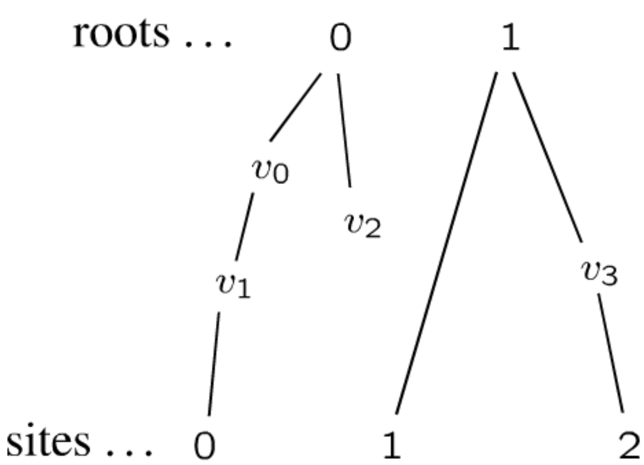
What's a Bigraph?

A **bigraph** *G* is a pair of graphs on the same (finite) set V of nodes:

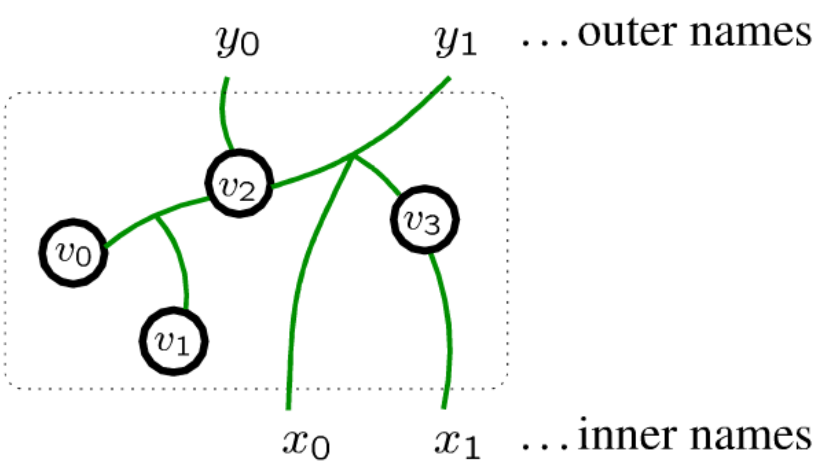
the **place graph**: a forest representing the topology of the system

the **link graph**: a hypergraph representing the connections

$G^{\mathrm{P}} : m \to n$

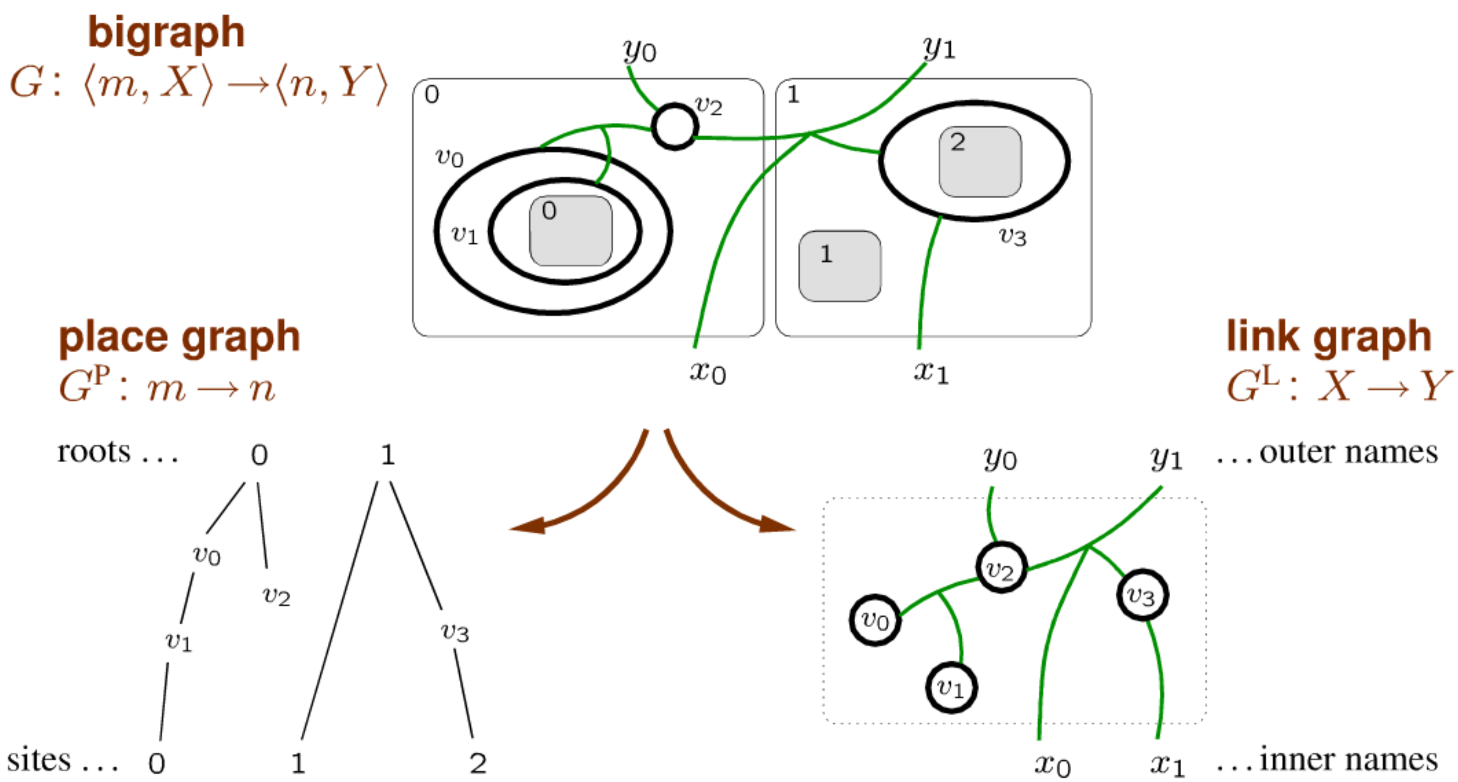


$G^{\mathrm{L}} : X \to Y$



What's a Bigraph?

Compact notation: place graph is represented by nesting nodes; sites are grey holes, and roots (or regions) are outlined

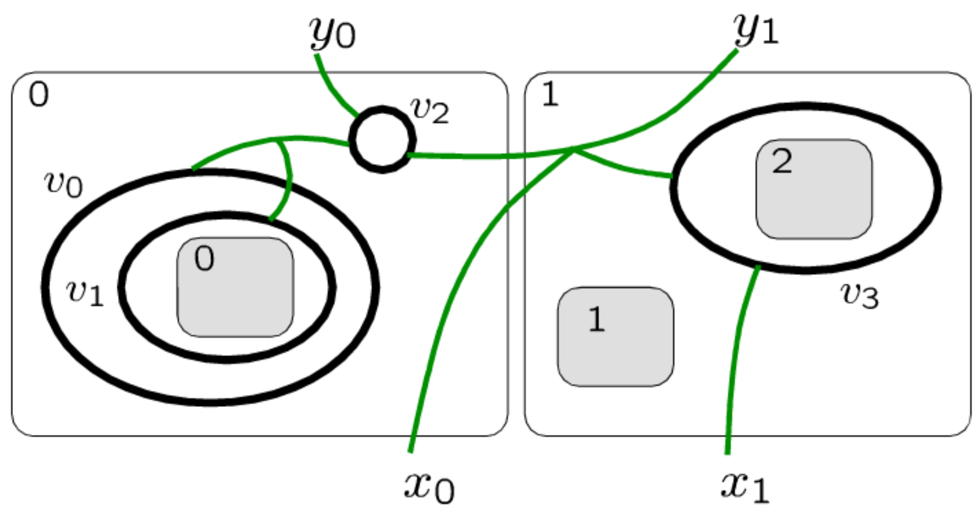


Each node is given a type (called **control**) taken from a set Σ called the **signature**.
The control specifies

- whether the node is *atomic*: atomic nodes must be empty
- the number of *ports* of the node

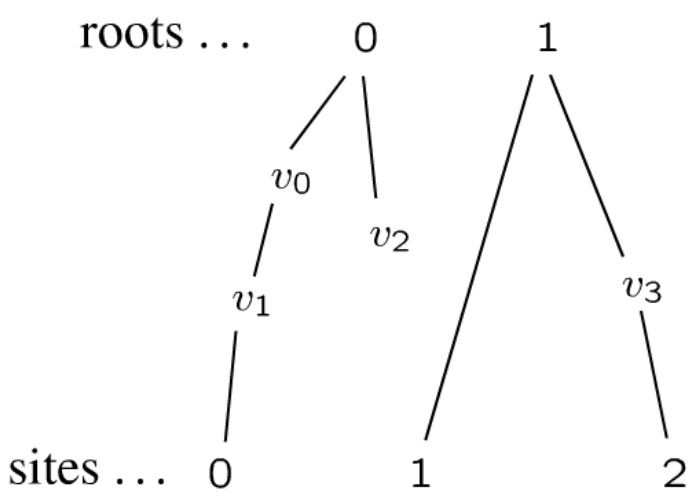
bigraph

$G : \langle m, X \rangle \to \langle n, Y \rangle$



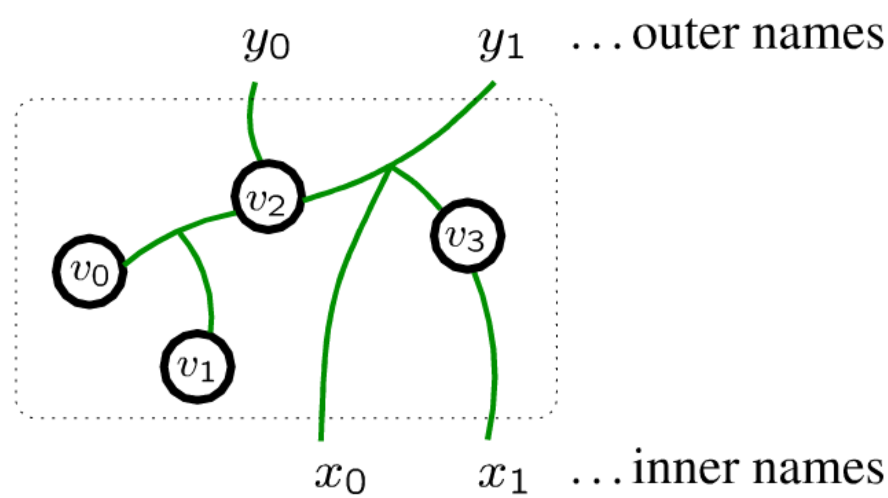
place graph

$G^{P} : m \to n$



link graph

$G^{L} : X \to Y$

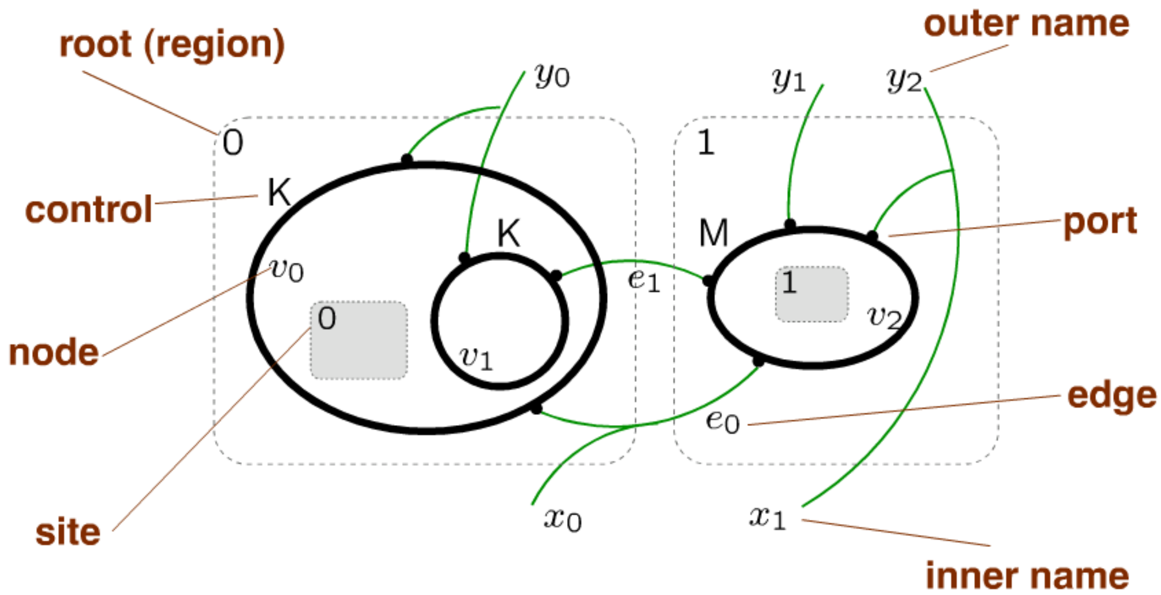


Each node is given a type (called **control**) taken from a set Σ called the **signature**.

The control specifies

- whether the node is *atomic*: atomic nodes must be empty
- the number of *ports* of the node

Anatomy of a Bigraph



- Bigraphs have a formal graphical language
- But complete textual languages (kind of graph grammars) are available
- "People want pictures. Coding is for nerds."

place = **root** or **node** or **site**

link = **edge** or **outer name**

point = **port** or **inner name**

$$G^P = (V, ctrl, prnt) \colon m \to n$$ **(place graph)**

$$G^L = (V, E, ctrl, edge, link) \colon X \to Y$$ **(link graph)**

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

$$= (G^P, G^L)$$

A pair <*m*,*X*> is an **interface**

- Bigraphs have a formal graphical language
- But complete textual languages (kind of graph grammars) are available
- "People want pictures. Coding is for nerds."

Horizontal composition of bigraphs (tensor)

Bigraphs can be juxtaposed when interfaces do not share names

Given $G_1 : \langle m_1, X_1 \rangle \to \langle n_1, Y_1 \rangle$, $G_2 : \langle m_2, X_2 \rangle \to \langle n_2, Y_2 \rangle$
the horizontal composition is

$$G_1 \otimes G_2 : \langle m_1 + m_2, X_1 \uplus X_2 \rangle \to \langle n_1 + n_2, Y_1 \uplus Y_2 \rangle$$

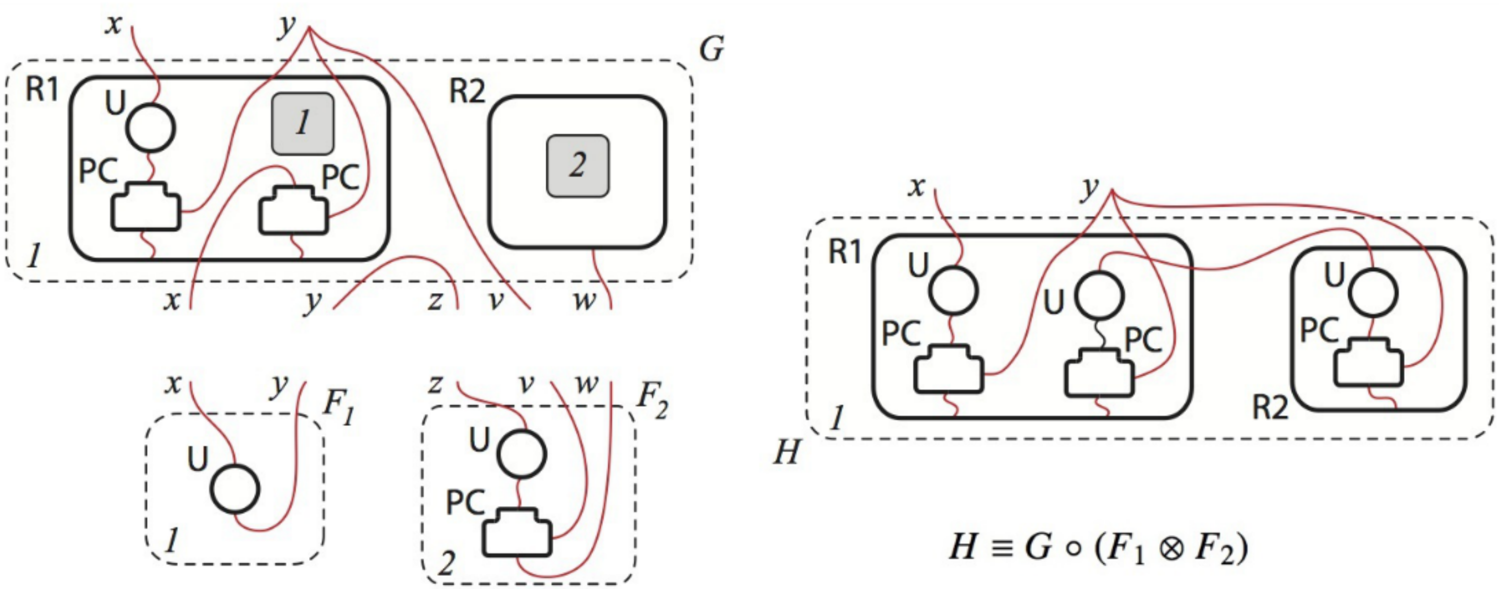
given by disjoint union of nodes, edges, and maps (possibly with renamings).
Notice that:

1. Order is important: $G_1 \otimes G_2 \neq G_2 \otimes G_1$
2. No links are added between the two summands
3. No roots are merged

Vertical Composition of bigraphs

Bigraphs can be composed when interfaces are compatible

- subbigraphs' roots are grafted in sites (holes)
- names are wired to same names



$$H \equiv G \circ (F_1 \otimes F_2)$$

(What is the identity for composition?)

Bigraphs form a monoidal category

Given a signature Σ, $(Big(\Sigma), \otimes, I)$ is the *monoidal category* where

- objects of $Big(\Sigma)$: interfaces
- morphisms of $Big(\Sigma)$: bigraphs over the signature Σ
- composition is vertical composition
- $\otimes$ is horizontal composition

(Categories of place graphs and link graphs can be defined likewise)

Categories of bigraphs are akin Lawvere theories

How to encode a process algebra - syntax

Basic strategy

Process Algebra	Bigraphs
syntactic constructor with n variables	control with arity n
syntactic tree	place graph
variable	outer name
name	edge

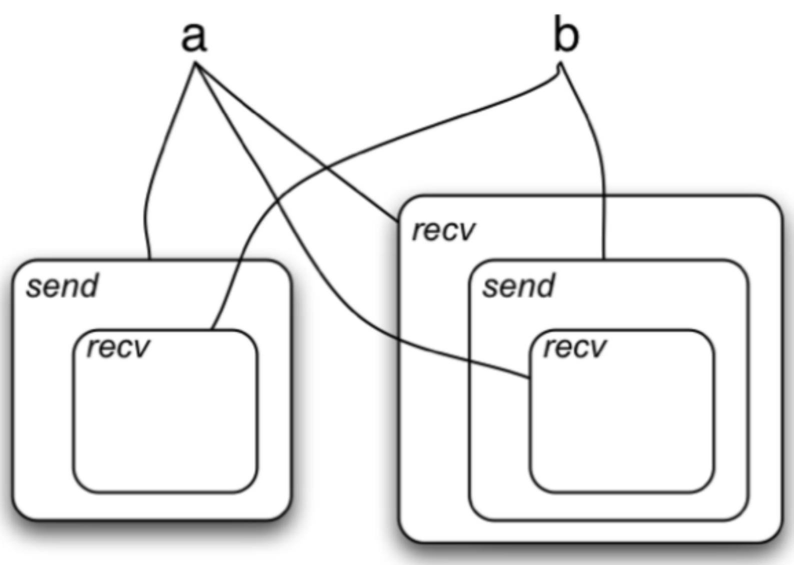
Useful shorthands: nil is just "nothing",
| (parallel) can be "omitted";

Example: CCS

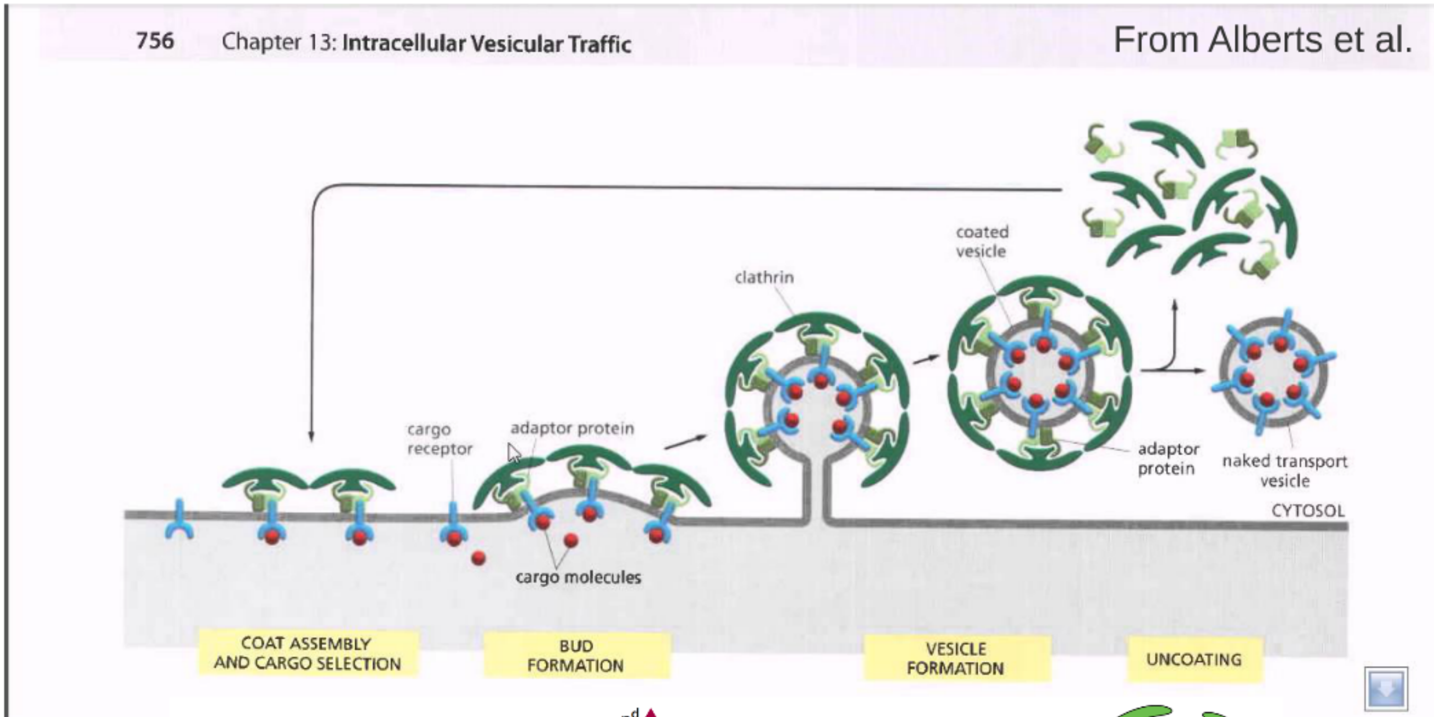
Original syntax: $P ::= 0 \mid a.P \mid \bar{a}.P \mid P|Q$

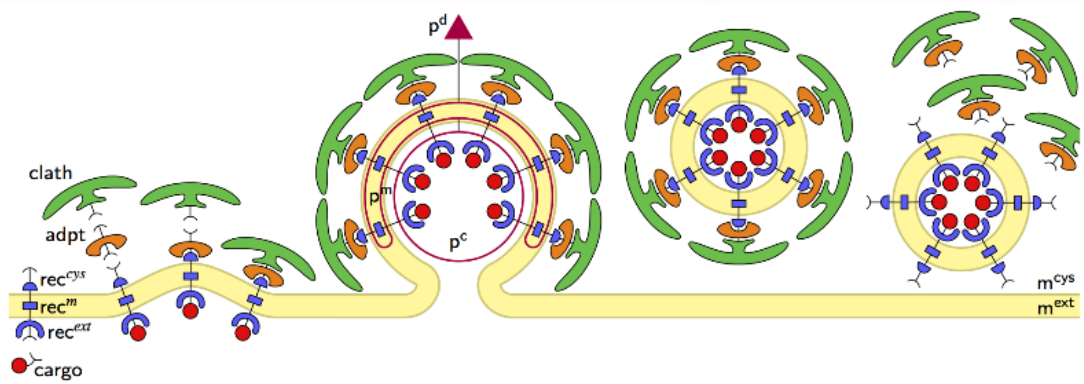
Bigraphs:
Signature has two controls: *send*, *recv*

Example: encoding of $\bar{a}.b.0|a.\bar{b}.a.0$



Modeling "informal" systems





This is a way to give a formal syntax to informal systems

Sortings

Often, bigraphs generated by a signature are too many.
Sorting = discipline for ruling out unwanted bigraphs

$$\frac{\text{sorted bigraphs}}{\text{unsorted bigraphs}} = \frac{\text{well-formed terms}}{\text{context-free terms}}$$

Can be specified in several ways (e.g. predicates in some logic); see work by Hildebrandt, Debois, Perrone, ...

Example: binding bigraphs

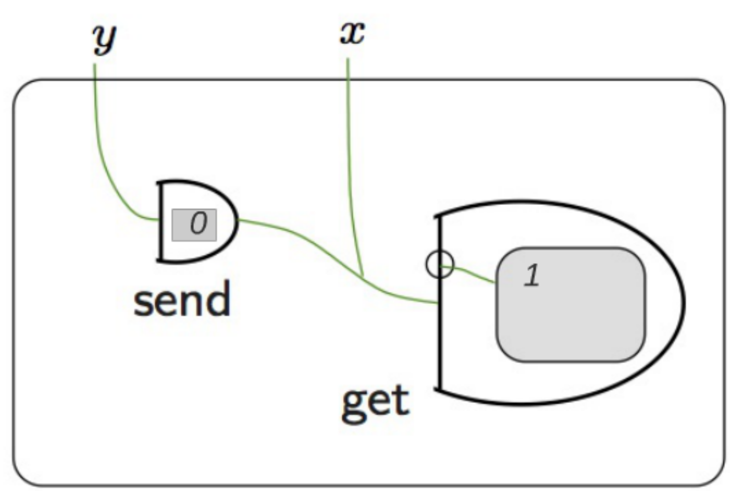
Some ports of controls are marked as *binding*

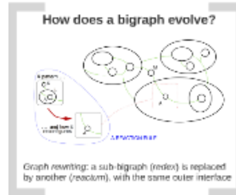
Requirement over bigraphs:

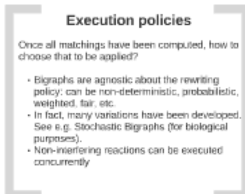
> "all points linked to a binding port of a node *u* lie inside *u*"

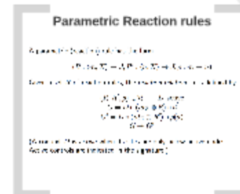
Example: encoding of π-calculus terms

$$\bar{x}y.P|x(z).Q$$

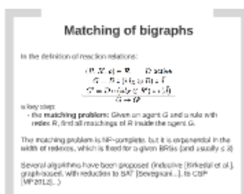


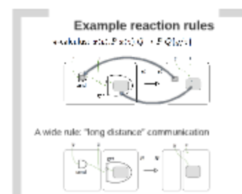

How does a bigraph evolve?
Graph rewriting: a sub-bigraph (redex) is replaced by another (reactum), with the same outer interface

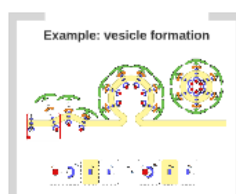

Execution policies
Once all matchings have been computed, how to choose that to be applied?
Bigraphs are agnostic about the rewriting policy: can be non-deterministic, probabilistic, weighted, fair, etc.
In fact, many variations have been developed. See e.g. Stochastic Bigraphs (for biological purposes).
Non-interfering reactions can be executed concurrently


Parametric Reaction rules

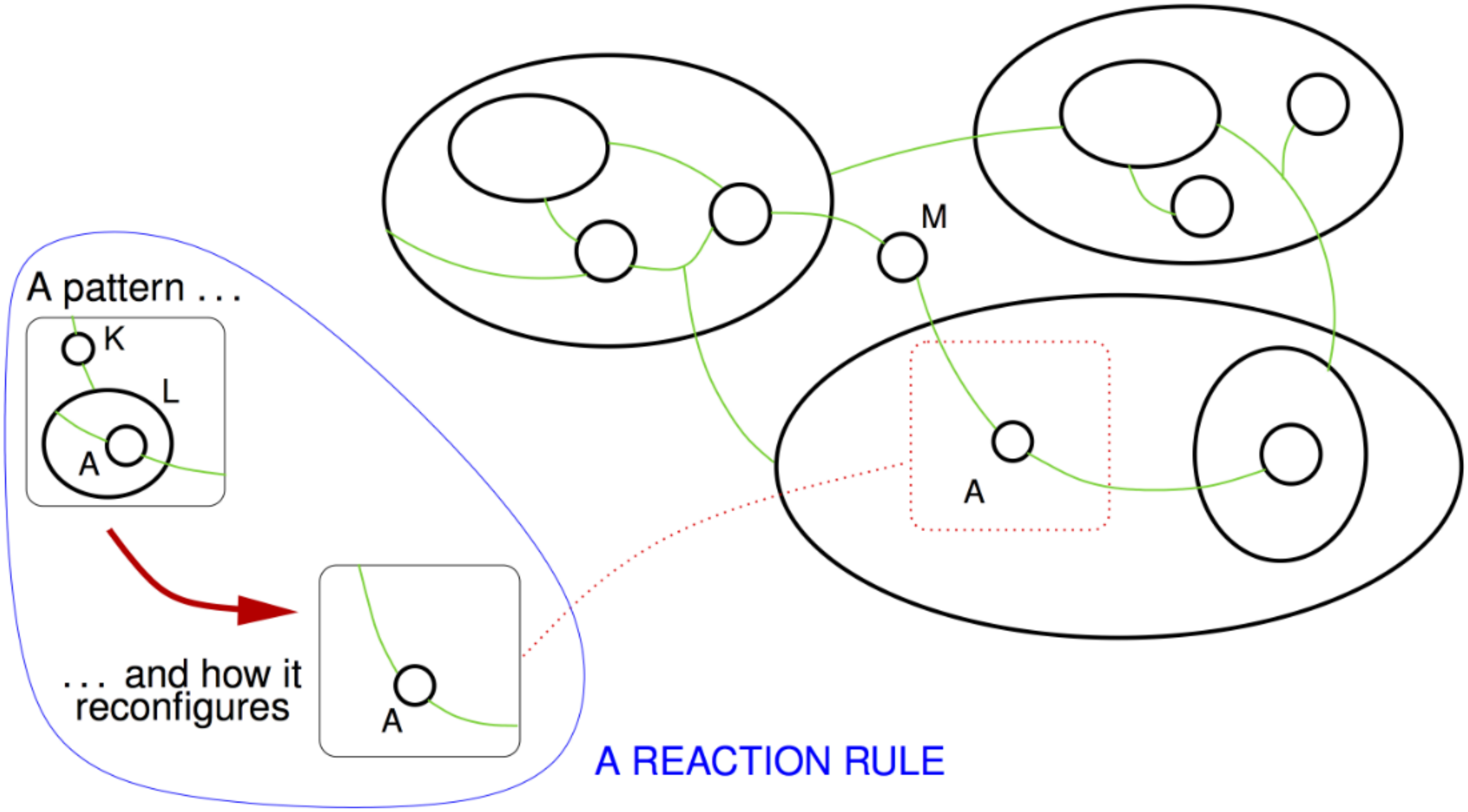
Dynamics


Matching of bigraphs


Example reaction rules
A wide rule: "long distance" communication


Example: vesicle formation

How does a bigraph evolve?



Graph rewriting: a sub-bigraph (*redex*) is replaced by another (*reactum*), with the same outer interface

Parametric Reaction rules

A parametric (reaction) rule has the form

$$(R : \langle m, X\rangle \to J, R' : \langle n, X\rangle \to J, \rho : m \to n)$$

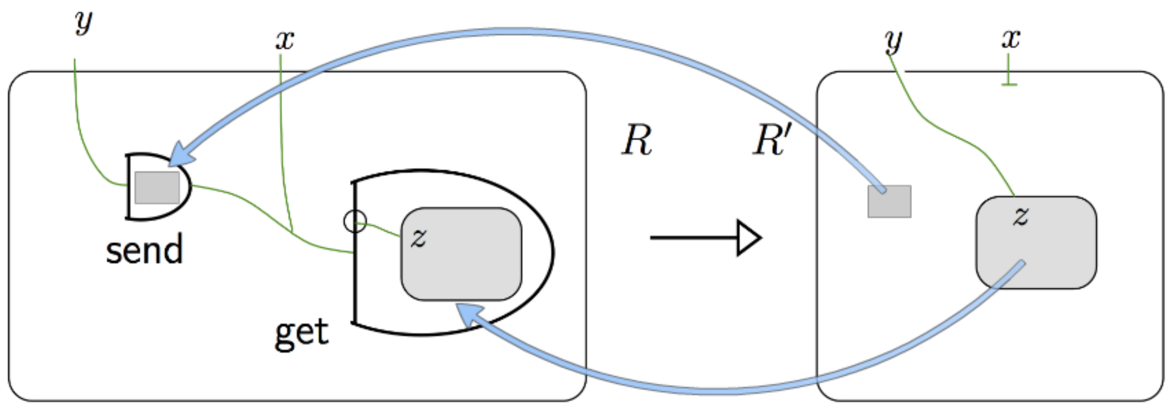
Given a set $\mathcal{R}$ of reaction rules, the *reaction relation* $\longrightarrow$ is defined by

$$\frac{\begin{array}{c}(R, R', \rho) \in \mathcal{R} \qquad D \text{ active} \\ G = D \circ (id_Z \otimes R) \circ \vec{d} \\ G' = D \circ (id_Z \otimes R') \circ \rho(\vec{d})\end{array}}{G \longrightarrow G'}$$

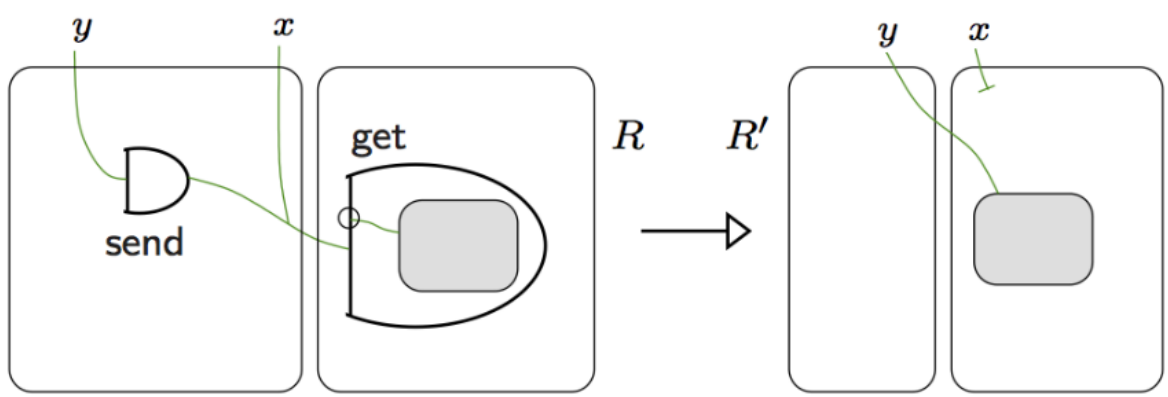
(A context D is *active* when its sites are only below active nodes. Active controls are indicated in the signature.)

Example reaction rules

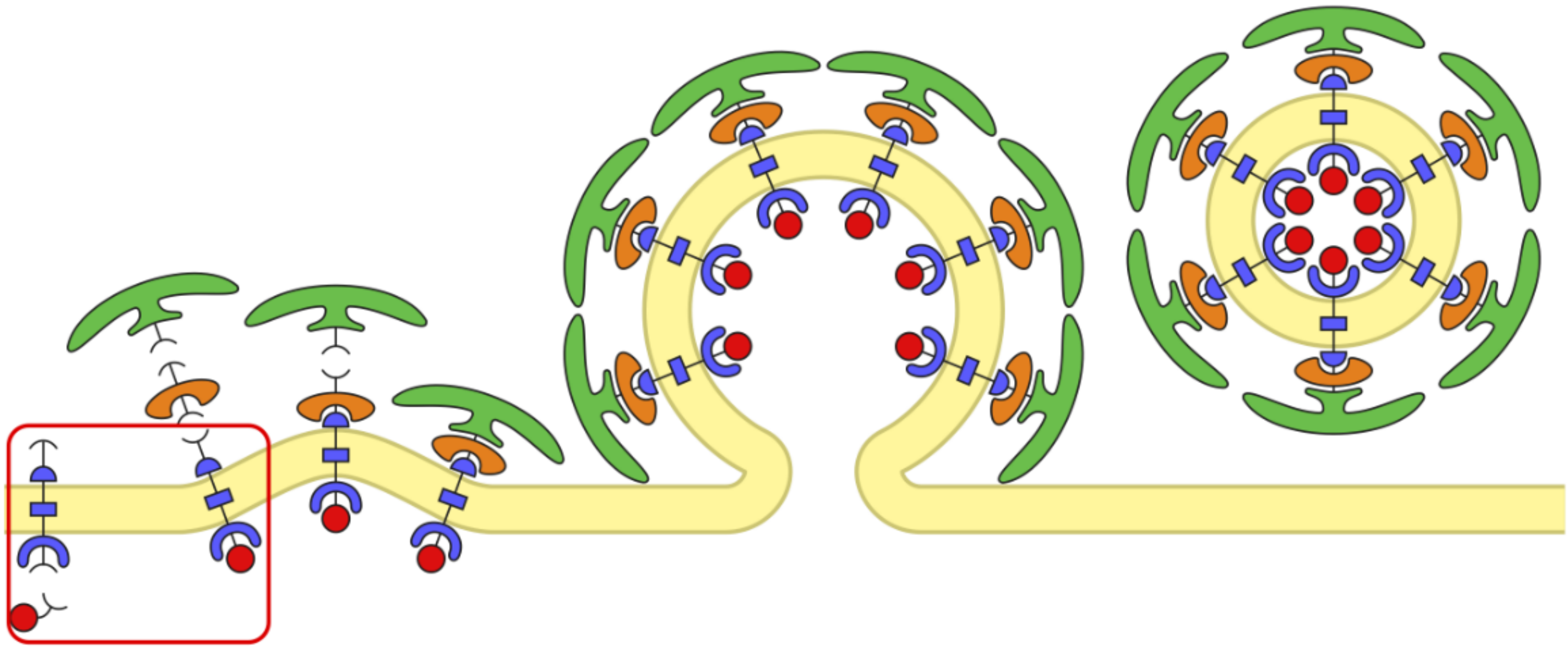
π-calculus: $x\langle y\rangle.P|x(z).Q \rightarrow P|Q\{y/z\}$

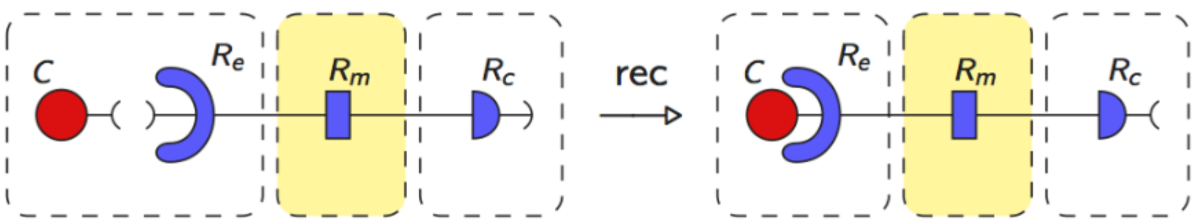


A wide rule: "long distance" communication



Example: vesicle formation





Matching of bigraphs

In the definition of reaction relations:

$$\frac{(R, R', \rho) \in \mathcal{R} \qquad D \text{ active} \\ G = D \circ (id_Z \otimes R) \circ \vec{d} \\ G' = D \circ (id_Z \otimes R') \circ \rho(\vec{d})}{G \rightarrowtriangle G'}$$

a key step:

- the **matching problem:** Given an agent *G* and a rule with redex *R*, find all matchings of *R* inside the agent *G*.

The matching problem is NP-complete, but it is exponential in the width of redexes, which is fixed for a given BRSs (and usually ≤ 3)

Several algorithms have been proposed (inductive [Birkedal et al.], graph-based, with reduction to SAT [Sevegnani...], to CSP [MP2012]...)

Execution policies

Once all matchings have been computed, how to choose that to be applied?

- Bigraphs are agnostic about the rewriting policy: can be non-deterministic, probabilistic, weighted, fair, etc.
- In fact, many variations have been developed. See e.g. Stochastic Bigraphs (for biological purposes).
- Non-interfering reactions can be executed concurrently

Metatheory & Tools

Deriving a good LTS

Often the semantics of a process algebra is given by means of a Labelled Transition System

$$P \xrightarrow{\alpha} Q$$

Useful for defining bisimilarity, model checking, etc.

Problem: how to define a good LTS for a process algebra?
Good = it induces a *compositional* bisimilarity

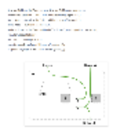
$$P \sim Q \text{ iff for all } C[\,]: C[P] \sim C[Q]$$

In general, it is a difficult and error-prone task.
(*cf.* the LTS for mobile ambients)

LibBig: a Java Library for Bigraphs

An implementation of the machinery for defining and manipulating bigraphical reactive systems.
Matching is implemented as a CSP.
Easily extensible and adaptable to other variants

http://mads.dimi.uniud.it/wordpress/downloads/libbig/



Labels from contexts

In a BRS we can define labels for an agent as the *minimal* contexts (i.e. bigraphs) which are required to make a reaction

[illegible]

C[ ] is "what G is missing" to make a reaction

In bigraphs, minimality is formally given by the categorical notion of *idempotent pushout* (IPO).

Tools

Simulation tools:

- nondeterministic execution engines (BAM [Perrone et al.])
- stochastic engines (Gillespie-based [Danos, Krivine,...])
- distributed [Mansutti, Peressotti, M., 2014]

(based on various algorithms for solving matching problem)

Model checkers [Perrone 2012, MPM in progress]

Graphical editors [Faithful, Hildebrandt]

...

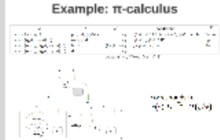
Labels from reactions

Theorem: in a BRSs, the bisimilarity given by the LTS whose labels are defined by IPOs. is always a congruence.

Hence, in order to get a LTS with a compositional bisimilarity for a process algebra:
- Encode the process algebra as a BRS
- Calculate the IPO labels

Often the resulting bisimilarity coincides with the knew one

Example: π-calculus



Proposition: The bisimilarity induced by IPOs coincides with the strong early bisimilarity.

Deriving a good LTS

Often the semantics of a process algebra is given by means of a Labelled Transition System

$$P \xrightarrow{a} Q$$

Useful for defining bisimilarity, model checking, etc.

Problem: how to define a good LTS for a process algebra?
Good = it induces a *compositional* bisimilarity

P ~ Q iff for all C[ ]: C[P] ~ C[Q]

In general, it is a difficult and error-prone task.
(cf. the LTS for mobile ambients)

Labels from contexts

In a BRS we can define labels for an agent as the *minimal* contexts (i.e. bigraphs) which are required to make a reaction

$$\frac{C[G] \rightarrowtriangle G' \quad C[\,] \text{ minimal}}{G \xrightarrow{C} G'}$$

C[ ] is "what *G* is missing" to make a reaction

In bigraphs, minimality is formally given by the categorical notion of *idempotent pushout* (IPO).

Labels from reactions

Theorem: in a BRSs, the bisimilarity given by the LTS whose labels are defined by IPOs, is always a congruence.

Hence, in order to get a LTS with a compositional bisimilarity for a process algebra:

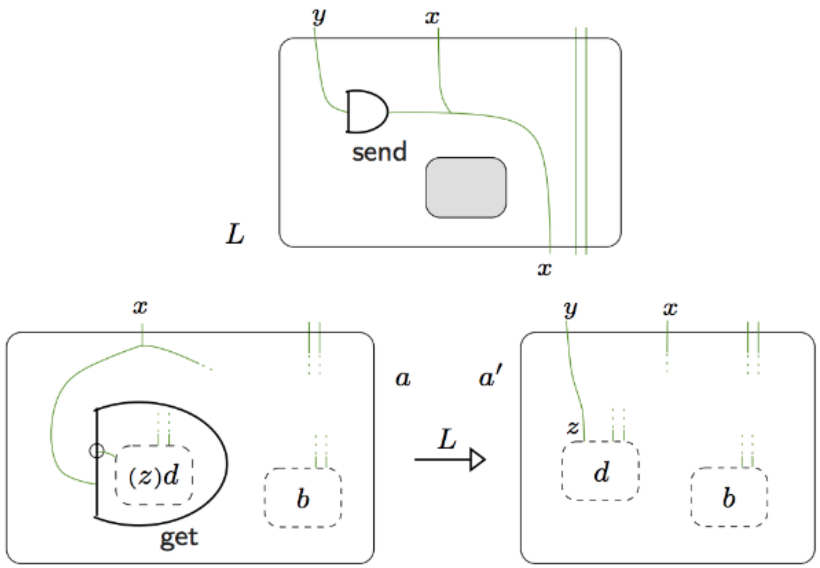
- Encode the process algebra as a BRS
- Calculate the IPO labels

Often the resulting bisimilarity coincides with the knew one

Example: π-calculus

a	L	a'	conditions	$\overline{L}$
$/W \circ (\mathsf{send}_{xy} \mid b)$	$\mathsf{get}_x \mid \mathsf{id},\ (z)d \otimes \mathsf{id}$	a'_0	$x \notin W,\ Z \cap (X \otimes W) = \emptyset,\ (z)d$ discrete	$\overline{x}(z)d_\pi$
$/W \circ ((\mathsf{get}_x \circ (z)d) \mid b)$	$\mathsf{send}_{xy} \mid \mathsf{id}$	a'_0	$x \notin W,\ y \notin X \otimes W$	xy
$/W \circ (\mathsf{send}_{xy} \mid (\mathsf{get}_u \circ (z)d) \mid b)$	$x/u \mid \mathsf{id}$	$(x/u \mid \mathsf{id}) \circ a'_0$	$x, u \notin W,\ x \neq u$	x/u
$/W \circ (\mathsf{send}_{xy} \mid (\mathsf{get}_x \circ (z)d) \mid b)$	id	a'_0		τ

where $a'_0 = /W \circ (x \mid (y/z \circ d) \mid b)$



corresponds to
$$x(z).D \xrightarrow{xy} D\{y/x\}$$

Proposition: The bisimilarity induced by IPOs coincides with the strong early bisimilarity.

Tools

Simulation tools:

- nondeterministic execution engines (BAM [Perrone et al.])
- stochastic engines (Gillespie-based [Danos, Krivine,...])
- distributed [Mansutti, Peressotti, M., 2014]

(based on various algorithms for solving matching problem)

Model checkers [Perrone 2012, MPM in progress]

Graphical editors [Faithful, Hildebrandt]

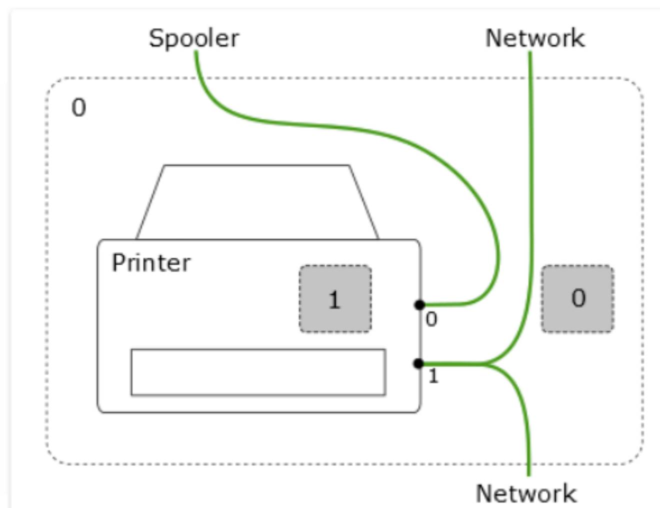
...

LibBig: a Java Library for Bigraphs

An implementation of the machinery for defining and manipulating bigraphical reactive systems.
Matching is implemented as a CSP.
Easily extensible and adaptable to other variants

http://mads.dimi.uniud.it/wordpress/downloads/libbig/

```
BigraphBuilder builder = new BigraphBuilder(signature);
OuterName spooler = builder.addOuterName("Spooler");
OuterName network = builder.addOuterName("Network");
Root root = builder.addRoot();
Node printer = builder.addNode("Printer",root,spooler,network);
builder.addSite(root);
builder.addSite(printer);
builder.addInnerName("Network",network);
Bigraph bigraph = builder.makeBigraph();
```



Bigraphs are a good operational metamodel

- Bigraphical Reactive Systems are a general operational meta-model which can be instantiated to many models and systems

- Provides a theory of general results and tools
- Graphically oriented, yet rigorously defined in category theory
- Many ideas have been ported to other contexts (e.g. IPOs are used in PROP categories)

- In fact, the "bigraphical way of thinking" is often used as a guideline in the design and analysis of distributed systems

Conclusions

Still to come...

- General BRSs analysis using Abstract Interpretation techniques (e.g. CFA, termination, interference...)
- Further development of library and tools
- CTL*-like spatial-temporal logic
- Applications (especially in agent-oriented programming)
- Overall, the model can evolve in different ways, so **feedback is very welcome!**

http://bigraph.org

Other cool stuff we had not time to see here

- Categorical formulations
- Application to barbed equivalence
- Agent-based programming [Pereira et al, Mansutti et al.]
- Programming languages refinement and engineering [Perrone et al, Grohmann et al]
- Variants (directed, stochastic, typed, etc.)
- Generalization: multi-graphs
- Computational bigraphs [Debois & Milner]
- Spatial logic (BiLog [Conforti et al])
- ...

Bigraphs are a good operational metamodel

- Bigraphical Reactive Systems are a general operational meta-model which can be instantiated to many models and systems

- Provides a theory of general results and tools
- Graphically oriented, yet rigorously defined in category theory
- Many ideas have been ported to other contexts (e.g. IPOs are used in PROP categories)

- In fact, the "bigraphical way of thinking" is often used as a guideline in the design and analysis of distributed systems

Other cool stuff
we had not time to see here

- Categorical formulations
- Application to barbed equivalence
- Agent-based programming [Pereira et al, Mansutti et al.]
- Programming languages refinement and engineering [Perrone et al, Grohmann et al]
- Variants (directed, stochastic, typed, etc.)
- Generalization: multi-graphs
- Computational bigraphs [Debois & Milner]
- Spatial logic (BiLog [Conforti et al])
- ...

Still to come...

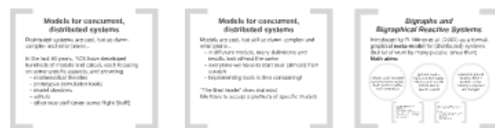
- General BRSs analysis using Abstract Interpretation techniques (e.g. CFA, termination, interference...)
- Further development of library and tools
- CTL*-like spatial-temporal logic
- Applications (especially in agent-oriented programming)
- Overall, the model can evolve in different ways, so **feedback is very welcome**!

http://bigraph.org

Bigraphical Reactive Systems

Marino Miculan
MADS lab
(with results by many people)

MeMo Workshop, June 6, 2014



Bigraphical Reactive System

A discrete *reactive system* is composed by a set of states and a transition relation.

A ***Bigraphical Reactive System*** is a RS where

- States are **bigraphs:** data structures rendering explicitly the positions and connections of system's components
- State transitions are **bigraph rewritings** defined by a set of local reaction rules

So BRSs propose as an *operational* metamodel.