Syntactic Theory for Bigraphs

Troels C. Damgaard (tcd@itu.dk)

Programming, Logics and Semantics (PLS) Group
IT University of Copenhagen
Denmark

Talk at Department of Informatics – University of Pisa
21st May 2007

Joint work with Lars Birkedal, Arne John Glenstrup, and Robin Milner
Relevant Publications


- Damgaard. *Syntactic Theory for Bigraphs*. Masters Thesis. IT University of Copenhagen, 2006. (Contains expanded introducing sections, proofs and related work in addition to the papers above.)

- And, an upcoming publication on the BPL implementation, based on the papers above.

See my webpage for more information — e.g., for links to more publications, slides, and to the Bigraphical Programming Languages group at ITU:

http://www.itu.dk/people/tcd/
Outline

1. Background — Bigraphs
2. Term Language
   - Expressing Bigraphs
   - Language
3. Normal Form Theorem(s)
4. Equational Theory
5. Characterization of Matching
   - Background — Bigraphical Reactive Systems
   - Matching Sentences
   - A First Step — Place Graph Matching Rules
   - Bigraph Matching Rules
6. Implementing Matching
   - Term Matching
   - A Prototype Implementation
7. Future Work
8. Appendix (Gritty details)
Introducing Bigraphs

Bigraphs and bigraphical reactive systems (BRSs)

In one line: A **graphical** model of **mobile computation** that emphasizes both **locality** and **connectivity** — due to Milner and coworkers.

Developed with two main aims

- to model **directly** important aspects of ubiquitous systems, and
- to provide a **general metatheory** for reactive systems, where we may represent and investigate many existing calculi for mobility and concurrency.

In particular, the theory provides us with tools to derive from a BRS a **labelled transition system** whose associated bisimulation relation is a congruence relation.
On the next few slides — a crash course on bigraphs.
We return to how to define bigraphical reactive systems.
On the next few slides — a crash course on bigraphs.

We return to how to define bigraphical reactive systems.

So what is a bigraph?

Essentially, a bigraph is

- a place graph — a forest, and
- a link graph — a (hyper-)graph sharing nodes.
Introducing bigraphs (cont.)

A bigraph — a combination of a place graph and a link graph

These are examples of place and link graphs:

But we like to draw a bigraph in a single picture, like this:

On the next few slides, an example of a concrete bigraph model. . . →
A consists of

- **roots** (dashed boxes),
- **nodes** (solid boxes), and
- **links** (green lines).

Each node has a **control** (in sans serif) indicating the number and type of **ports** for linkage.

Ports can be either **free** or **binding** — the latter indicated by circular attachments.
Bigraphs are Composable

Non-ground bigraphs contain holes (or sites) and/or inner names.

We can compose $B$ and $C$ by plugging the holes of $B$ with the roots of $C$. $B$ and $C$ compose to form $A$. We write $A = BC$. 
Binding bigraphs, enforce a scoping discipline on linkage connected to a binding ports or local outer names — binders.

All peers (inner names or ports) linked to a binder, must be nested within the node.
Formally, a bigraph is a morphism in a category of interfaces, hence, bigraph composition is simply the categorical composition.

For example, $B$ and $C$ compose exactly, because the innerface of $B$ is equal to the outerface of $C$.

We can also combine bigraphs with a tensor product, $\otimes$, which is simply juxtaposition of roots; requiring that outer and inner names are disjoint.

A parallel product $B \parallel C$, requiring only inner names to be disjoint can be straightforwardly derived from $\otimes$.
How to give a bigraph? Two basic approaches —

**Construct graph structure directly**

Bigraphs are simply a collection of certain graphs.

We can just give

- explicit sets of constituents (i.e., nodes, links, names, controls, etc.); and,
- a number of maps to build bigraph structure (i.e., nesting, linking, etc.).

**Construct inductively from set of elementary bigraphs**

Construct bigraphs inductively as the smallest set of bigraphs built from

- a set of elementary bigraphs; and,
- a few operators.
  - composition $G_0G_1$ — vertical composition of bigraphs,
  - product $G_0 \otimes G_1$ — horizontal composition of bigraphs,
  - and abstraction $(X)P$ — to make global names $X$ of $P$ local.
Language for Bigraphs (cont.)

Building an expression for $A$

$$\text{id}_1 \otimes / \{y_0, y_1\} \otimes \text{id}_1 \otimes / \{z_0, z_1\}$$
Building an expression for $A$

$$\text{server}\ y_0(x)$$

$$\text{id}_1 \otimes / \{y_0, y_1\} \otimes \text{id}_1 \otimes / \{z_0, z_1\}$$
Building an expression for $A$

$$\text{id}_1 \otimes /\{y_0, y_1\} \otimes \text{id}_1 \otimes /\{z_0, z_1\}$$

server$_{y_0}(x)$

$(x)$ secret$_x$
Language for Bigraphs (cont.)

Building an expression for $A$

$$\text{id}_1 \otimes / \{y_0, y_1\} \otimes \text{id}_1 \otimes / \{z_0, z_1\}$$

$\text{server}_{y_0(x)}$

$(x) \text{secret}_x$

$1$
Language for Bigraphs (cont.)

Building an expression for $A$

$$\text{id}_1 \otimes / \{y_0, y_1\} \otimes \text{id}_1 \otimes / \{z_0, z_1\}$$

$$y_1/y_1 \otimes \text{office} \otimes z_0/z_0 \otimes z_1/z_1$$

server$_{y_0}(x)$

(secret$_x x$)

1
Language for Bigraphs (cont.)

Building an expression for $A$

$\text{id}_1 \otimes / \{y_0, y_1\} \otimes \text{id}_1 \otimes / \{z_0, z_1\}$

$y_1/y_1 \otimes \text{office} \otimes z_0/z_0 \otimes z_1/z_1$

$y_1/y_1 \otimes \text{merge}_3 \otimes z_0/z_0 \otimes z_1/z_1$
Language for Bigraphs (cont.)

Building an expression for $A$

$$id_1 \otimes / \{y_0, y_1\} \otimes id_1 \otimes / \{z_0, z_1\}$$

$$y_1/y_1 \otimes \text{office} \otimes z_0/z_0 \otimes z_1/z_1$$

$$y_1/y_1 \otimes \text{merge}_3 \otimes z_0/z_0 \otimes z_1/z_1$$

$$pc_{y_1} \otimes \text{pda}_{z_0} \otimes \text{pda}_{z_1}$$
Language for Bigraphs (cont.)

Building an expression for $A$

$$\begin{align*}
\text{id}_1 \otimes /\{y_0, y_1\} \otimes \text{id}_1 \otimes /\{z_0, z_1\} \\
y_1/y_1 \otimes \text{office} \otimes z_0/z_0 \otimes z_1/z_1 \\
y_1/y_1 \otimes \text{merge}_3 \otimes z_0/z_0 \otimes z_1/z_1 \\
\text{pc}_{y_1} \otimes \text{pda}_{z_0} \otimes \text{pda}_{z_1} \\
1 \otimes 1 \otimes 1
\end{align*}$$
Conclusion
Completeness of Language for Binding Bigraphs

Theorem (Completeness of Language)

All binding bigraphs can be expressed using composition, tensor product \( \otimes \), and abstraction \((\ )\), from constants

\[ 1, \text{merge}_2, y/X, /x, K_{\vec{y}(\vec{x})}, \pi, \lceil U \rceil. \]

...follows immediately from the normal form theorem.
Motivation

As a basis for structured analysis of bigraphs — we develop a normal form theorem that

- details a number of subclasses of bigraphs (overview, next slide); and,
- gives a corresponding expression format for each class.
We analyze bigraphs and define

- discrete decomposition — separating a bigraph $B$ into a global link graph $w$ and a discrete bigraph $D$, which has one-one linkage to global outer names;

\[ B = \begin{array}{c}
\text{id} \\
\end{array} \quad \begin{array}{c}
w \\
\end{array} \quad \begin{array}{c}
D \\
\end{array} \]
Normal Forms (cont.)
The discrete variant for matching

We analyze bigraphs and define

- **discrete** decomposition — separating a bigraph $B$ into a **global** link graph $w$ and a **discrete** bigraph $D$, which has one-one linkage to global outer names;

- decomposing a discrete bigraph $D$ into separate roots — discrete primes $P$; and,
Normal Forms (cont.)

The discrete variant for matching

We analyze bigraphs and define

- **discrete** decomposition — separating a bigraph $B$ into a **global** link graph $w$ and a **discrete** bigraph $D$, which has one-one linkage to global outer names;
- decomposing a discrete bigraph $D$ into separate roots — discrete **primes** $P$; and,
- decomposing discrete primes $P$ into separate nodes and holes.

![Diagram of bigraphs and decomposition](image)
Main theorem states soundness and completeness of the normal form;
Conclusion

Main result - soundness and completeness of the normal form

Main theorem states soundness and completeness of the normal form; and takes about two pages to state in full detail.

**Theorem (Schema: Normal Form)**

*Any bigraph of class $C$ can be expressed on the format $E$. For any other expression $E'$ on this format, the requirements $R_1, \ldots, R_{n-1}$ (between constituents of $E$ and $E'$) hold. (Formats are unique up to certain isos.)*
Outline

1. Background — Bigraphs
2. Term Language
   - Expressing Bigraphs
   - Language
3. Normal Form Theorem(s)
4. Equational Theory
5. Characterization of Matching
   - Background — Bigraphical Reactive Systems
   - Matching Sentences
   - A First Step — Place Graph Matching Rules
   - Bigraph Matching Rules
6. Implementing Matching
   - Term Matching
   - A Prototype Implementation
7. Future Work
8. Appendix (Gritty details)
Motivation

Would like to reason about graph equality on the term level — i.e., to reason syntactically about equality of bigraphs.

For example, we might like to show that:

\[(p \circ y_1 \otimes p \circ d a_{z_0}) \cdot (1 \otimes 1) = (p \circ y_1 \cdot 1 \otimes p \circ d a_{z_0} \cdot 1),\]

\[y/x \cdot x/z = y/z.\]

We give a syntactic equational theory by stating basic equalities between bigraph expressions — extending work for pure bigraphs by Milner.
Categorical axioms

Categorical axioms deal with basic properties like

- associativity of composition and tensor product, $A(BC) = (AB)C$ and $A \otimes (B \otimes C) = (A \otimes B) \otimes C$,
- unit for composition and tensor products (identities on certain interfaces),
- and handling symmetries (permutations on the place graph structure)

\[(C1) \quad A \ id_I = A = id_J A \quad (A : I \to J)\]
\[(C2) \quad A(BC) = (AB)C\]
\[(C3) \quad A \otimes id_\epsilon = A = id_\epsilon \otimes A\]
\[(C4) \quad A \otimes (B \otimes C) = (A \otimes B) \otimes C\]
\[(C5) \quad id_I \otimes id_J = id_{I \otimes J}\]
\[(C6) \quad (A_1 \otimes B_1)(A_0 \otimes B_0) = (A_1A_0) \otimes (B_1B_0)\]
\[(C7) \quad \gamma_{I,\epsilon} = id_I\]
\[(C8) \quad \gamma_{J,I}\gamma_{I,J} = id_{I \otimes J}\]
\[(C9) \quad \gamma_{I \otimes J,K} = (\gamma_{I,K} \otimes id_J)(id_I \otimes \gamma_{J,K})\]
\[(C10) \quad \gamma_{I,K}(A \otimes B) = (B \otimes A)\gamma_{H,J} \quad (A : H \to I, B : J \to K)\]
Global link axioms

Link axioms mainly state the ways in which compositions of linkings can be equally expressed without composition.

(L1) \( x/x = \text{id}_x \)
(L2) \( /y y/x = /x \)
(L3) \( /y y = \text{id}_\epsilon \)
(L4) \( z/\{Y \uplus y\}(\text{id}_Y \otimes y/X) = z/\{Y \uplus X\} \)
Placing and ion axioms

Place axioms deal with unit, associativity and commutativity of the placing elementary bigraphs.

(P1) \( \text{merge}_2(1 \otimes \text{id}_1) = \text{id}_1 \)
(P2) \( \text{merge}_2(\text{merge}_2 \otimes \text{id}_1) = \text{merge}_2(\text{id}_1 \otimes \text{merge}_2) \)
(P3) \( \text{merge}_2 \gamma_{1,1,\emptyset,\emptyset} = \text{merge}_2 \)

Two axioms for ions deal with renaming of the names on the inner and outer interface of an ion.

(N1) \( (\text{id}_1 \otimes \alpha)K_{\tilde{y}(\tilde{x})} = K_{\alpha(\tilde{y})}(\tilde{x}) \)
(N2) \( K_{\tilde{y}(\tilde{x})}^{\tilde{\sigma}} = K_{\tilde{y}(\tilde{\sigma}^{-1}(\tilde{x}))} \)
Binding axioms

Binding axioms state basic equalities between expressions with the abstraction operator (locating names) and/or the concretion constant (globalizing names).

\[(\emptyset)P = P\] locating no names is the same as not locating.

\[(Y)(\downarrow Y \uparrow) = \text{id}(Y)\] (re)locating names Y made global by a concretion is the same as leaving them local.

\[(\uparrow X \downarrow Z \otimes \text{id}_Y)(X)P = P\] locating names X of P, and then globalizing these names again, leaves P unchanged.

\[(\text{id}_X \otimes (Y)P)G = (Y)(P \otimes \text{id}_X)G\] we are allowed to extend the scope of an abstraction (requiring only that abstraction be welldefined).

\[(X \uplus Y)P = (X)(Y)P\] locating names Y and names X is the same as locating the union of those names.
Conclusion

Main result - soundness and completeness of the equational theory

Theorem (Soundness and completeness)

For all binding bigraph expressions $E$ and $F$, $[E] = [F]$, i.e., $E$ and $F$ denote the same bigraph iff $\vdash E = F$. 
Outline

1. Background — Bigraphs
2. Term Language
   - Expressing Bigraphs
   - Language
3. Normal Form Theorem(s)
4. Equational Theory
5. Characterization of Matching
   - Background — Bigraphical Reactive Systems
   - Matching Sentences
   - A First Step — Place Graph Matching Rules
   - Bigraph Matching Rules
6. Implementing Matching
   - Term Matching
   - A Prototype Implementation
7. Future Work
8. Appendix (Gritty details)
We build **bigraphical reactive systems** (BRS) by giving a set of rewriting rules; expressed essentially as a pair of bigraphs, like those below:

We can rewrite a bigraph $a$ with a rule $R \Rightarrow R'$, if $a$ matches $R$. 
The matching problem

To determine, whether and how a redex matches a bigraph.

Suppressing some detail — a redex $R$ matches a ground agent $a$, if $a$ decomposes, s.t.,

$$a = C(R \otimes \text{id}_Z) d$$

— for context $C$, and discrete parameter $d$. We can illustrate a match schematically, like this:
Motivation
— creating a bigraph manipulation tool proven correct in great detail

- A bigraphical reactive system (BRS) is a system state and a set of reaction rules specifying how the state can change.
- In the BPL group, we would like to create a tool for experimenting with bigraphs.
- The first step is building an engine for simulating BRSs.
- We would like it to be proven correct in as much detail as possible.
- Matching is a core problem.
Motivation (cont.)

So, we have definitions

Definition (A bigraph)

“Official” definition: \( G = (V, E, ctrl, link, prnt) : I \rightarrow J \)

Definition (A match in bigraphs)

“Official” definition: […] \( a = C(R \otimes \text{id}_Z) d \), where \( C \) is a context, and \( d \) a discrete parameter […]

Problem (Constructing a context and parameter)

How to construct a context \( C \) and parameter \( d \), given agent \( a \) and redex \( R \)?

Instead, we

- look to characterize in a constructive manner the matching problem; and,
- work to specialize the characterization into an algorithmic approach.
Define a new representation for a match.
We simply decompose discretely agent, context, redex and parameter.

Matching sentence for binding bigraphs

A matching sentence for binding bigraphs is a 7-tuple relation

\[ \omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d \]

where \( \omega_a, \omega_R, \omega_C \) are wirings, and \( a, R, C, d \) are discrete.

It is valid iff

\[
(id \otimes \omega_a)a = (id \otimes \omega_C)(id_{Z \uplus V} \otimes C)(id_Z \otimes (id \otimes \omega_R)R)d.
\]
Rules for Deriving Valid Matching Sentences

Matching rules

We give a set of rules for deriving valid matching sentences.

Intuition:

1. We start with a set of rules for deriving valid place graph sentences.
2. We augment the rules with (local and global) wiring to infer valid matching sentences for bigraphs.
Inference System for Place Graph Matches

Step one — inferring valid place graph sentences

Place graphs are built from product and composition from:

- \( \text{merge}_n, \pi \) and \( K \).
- Restricted to place graphs matching sentences are simply:
  \[ a, R \leftrightarrow C, d \]

\( a = CRd \).
General Inference Tree Structure
— match redex above, context below SWITCH

- PERM, MERGE and ION rules match agent and context structure.
- SWITCH moves the redex into context position.
- Between any leaf and the root, SWITCH is applied at most once.

Intuitive interpretation of the SWITCH rule

Rules applied above SWITCH match agent and redex structure.
Rules applied below SWITCH match agent and context structure.
Inference System for Binding Bigraph Matching

Step two (a) — augmenting inference rules with wiring constructs

**PERM**

\[
\omega_a, \omega_R, \omega_C \vdash a, \bigotimes_i^m P_{\pi^{-1}(i)} \leftrightarrow C, (\pi \otimes \text{id})d
\]

\[
\omega_a, \omega_R, \omega_C \vdash a, \bigotimes_i^m P_i \leftrightarrow C_{\pi}, d
\]

**MERGE**

\[
\omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d \quad \text{a global}
\]

\[
\omega_a, \omega_R, \omega_C \vdash (\text{merge} \otimes \text{id})a, R \leftrightarrow (\text{merge} \otimes \text{id})C, d
\]

**PRODUCT**

\[
\omega_a, \omega_R, \omega_C \mid \omega \vdash a, R \leftrightarrow C, d
\]

\[
\omega_b, \omega_S, \omega_D \mid \omega \vdash b, S \leftrightarrow D, e
\]

\[
\omega_a \parallel \omega_b, \omega_R \parallel \omega_S, \omega_C \parallel \omega_D \mid \omega \vdash a \otimes b, R \otimes S \leftrightarrow C \otimes D, d \otimes e
\]

**ION**

\[
\omega_a, \omega_R, \omega_C \vdash ((\vec{v})/(\vec{X}) \otimes \text{id})p, R \leftrightarrow ((\vec{v})/(\vec{Z}) \otimes \text{id})P, d
\]

\[
\sigma \parallel \omega_a, \omega_R, \sigma \alpha \parallel \omega_C \vdash (K_{\vec{y}(\vec{X})} \otimes \text{id})p, R \leftrightarrow (K_{\vec{u}(\vec{Z})} \otimes \text{id})P, d
\]

\[
\alpha = \vec{y}/\vec{u} \quad \sigma : \{\vec{y}\} \rightarrow
\]

**SWITCH**

\[
\omega_a, \text{id}_\epsilon, \omega_C(\sigma \otimes \omega_R \otimes \text{id}) \vdash p, \text{id} \leftrightarrow P, d
\]

\[
P : \rightarrow \langle W \uplus Y \rangle \quad \sigma : W \rightarrow U
\]

\[
\omega_a, \omega_R, \omega_C \vdash p, (\hat{\sigma} \otimes \text{id})(W)P \leftrightarrow \Gamma U, d
\]

**PAX**

\[
\sigma : W \uplus U \rightarrow \alpha : V \rightarrow W \quad \beta : Z \rightarrow U \quad \tau : X \rightarrow V
\]

\[
p : \langle X \uplus Z \rangle
\]

\[
\sigma(\beta \otimes \alpha \tau), \text{id}_\epsilon, \sigma \vdash p, \text{id}_{(V)} \leftrightarrow \Gamma \alpha, (\beta \otimes \hat{\tau})(X)p
\]
Inference System for Binding Bigraph Matching

Step two (b) — adding rules for handling wiring constructs

3 rules are added, to handle

- the base case for bigraphs that are just wirings.
- making local wiring global, when removing an abstraction.
- opening closed links (nonlocated edges).

**WIRING-AXIOM**

\[ y, X, y/X \vdash \text{id}_\epsilon, \text{id}_\epsilon \leftarrow \text{id}_\epsilon, \text{id}_\epsilon \]

**ABSTR**

\[
\sigma_a \otimes \omega_a, \omega_R, \sigma_C \otimes \omega_C \vdash p, R \leftarrow P, d \\
\sigma_a : Z \rightarrow W \quad \sigma_C : U \rightarrow W \\
P : U \uplus X \\
\omega_a, \omega_R, \omega_C \vdash (\hat{\sigma}_a \otimes \text{id})(Z) p, R \leftarrow (\hat{\sigma}_C \otimes \text{id})(U) P, d \\
\]

**CLOSE**

\[
\sigma_a, \sigma_R, \text{id}_{Y_R} \otimes \sigma_C \vdash a, R \leftarrow C, d \\
\sigma_C : \rightarrow Y \uplus Y_C \\
\sigma_R : \rightarrow U \uplus Y_R \\
(id \otimes / (Y_R \uplus Y_C)) \sigma_a, (id \otimes / Y_R) \sigma_R, (id \otimes / Y_C) \sigma_C \vdash a, R \leftarrow C, d \\
\]

Troels C. Damgaard (ITU)
Conclusion
Main result — soundness and completeness of the rules for binding bigraphs

Theorem (Soundness)

The rules for matching are sound, that is, any matching sentence that can be derived is valid.

Theorem (Completeness)

The rules for matching are complete; that is, we can infer

\[ \omega_a, \omega_R, \omega_C \vdash a, R \hookrightarrow C, d \]

exactly when it is valid, i.e.,

\[ (\text{id} \otimes \omega_a)a = (\text{id} \otimes \omega_C)(\text{id}_{Z \uplus V} \otimes C)(\text{id}_Z \otimes (\text{id} \otimes \omega_R)R)d. \]

The equality theory and normal forms are employed heavily to prove these theorems.
In the BPL tool, bigraphs are given and represented using a term language with constants and operators corresponding exactly to the elementary bigraphs and combinators.

We can implement matching \( a = C(id_Z \otimes R)d \) by

Implementing matching

1. decomposing \( a \) and \( R \) discretely (as \( a = (id \otimes \omega_a)a' \) and \( R = (id \otimes \omega_R)R' \))
2. inferring a match \( \omega_a, \omega_R, \omega_C \vdash a', R' \hookrightarrow C', d \)
3. computing \( C = (id \otimes \omega_C)(id_{Z \uplus V} \otimes C') \)

Problem: \( a, R, C, d \) in rules are bigraphs, not bigraph terms.

In other words, to match bigraphs given as terms we need to add a rule allowing us to apply structural congruence.

Instead of adding such a rule, we
   - add a single rule DNF to rewrite a term to normal form; and,
   - incorporate structural congruence axioms into PRODUCT and MERGE rules.

This is proved to be enough to reformulate the rules over terms.
Normal Inferences for Term Matching

Normal inferences

To specialize the characterization into an algorithm, for mechanically finding matches, define normal inferences; kinds of inferences that are

- **complete** in the sense that all valid matching sentences can be inferred;
- suitable **restricted**, s.t. inferences can be built mechanically with minimum amount of search; and,
- in particular, normal inference definitions for term matching must address precisely how and where to apply structural congruence (with the help of the DNF rule and augmented PRODUCT and MERGE rules).

A point on normal inferences vs. algorithmic approach

Each definition of normal inferences correspond to a different algorithmic approach to matching.
Based on the matcher engine, we have made a command line interface, running as an extension of, e.g. the SMLNJ interactive command line:

```
- val K = active0 "K";
val K = K : 1 -> 1 : bgval
- val L = active0 "L";
val L = L : 1 -> 1 : bgval
- print_mv(match_v{agent = K o merge(2) o (L o <-> * K o <->),
  redex = K}) handle e => explain e;
[{context = idp(1), parameter = merge(2) o (L o <-> * K o <->)},
{context = K o merge(2) o (L o <-> * ‘[]‘), parameter = <->}]
val it = () : unit
```

See [http://www.itu.dk/research/bpl/](http://www.itu.dk/research/bpl/).
BPLweb

Agent:  
K 0 merge(2) 0 (L 0 <-> * K 0 <->)

Rules:

Rule 0:
Redex:  
K
Reactum:  
M
Instantiation:  
[0 |-> 0]

Matches:

Match 0:
Context:

idp(1)

Parameter:
merge(2) 0 (L 0 <-> * K 0 <->)

Inference tree:

Match 1:
Context:

K 0 merge(2) o (L 0 <-> * `[]`)

Parameter:
<->

Inference tree:
We have implemented a BRS tool proven correct in great detail. Implementation strictly adheres to formal rules. Key techniques include

- characterizing matching inductively — as a set of sentences inferable by a set of rules,
- recasting matching sentences over bigraph terms
- — with the help of an axiomatization of structural congruence; and
- normal forms for bigraphs.

Command line tool and web demo front ends are available!

http://www.itu.dk/research/bpl/
Outline

1. Background — Bigraphs
2. Term Language
   • Expressing Bigraphs
   • Language
3. Normal Form Theorem(s)
4. Equational Theory
5. Characterization of Matching
   • Background — Bigraphical Reactive Systems
   • Matching Sentences
   • A First Step — Place Graph Matching Rules
   • Bigraph Matching Rules
6. Implementing Matching
   • Term Matching
   • A Prototype Implementation
7. Future Work
8. Appendix (Gritty details)
Future Work

On matching

Term-based matching using normal inferences is

- **good** — it is firmly based on the theory presented here; and
- **problematic** — matching in the pilot-version is quite inefficient.

Hence, we aim (and hope to be able) to build more efficient methods; while retaining our formal foundation.

Some ideas:

- Work with a **term normal form** more suited for matching.
- Instead of matching links “online” while building an inference, derive a set of **constraints** for these.
- Pre-computing **approximations** of possible matching locations.
- **Add rules** to allow shortcutting obviously invalid/valid matches.
- Smarter ways of **combining matching and rewriting**.
Thank you for listening!
Outline of Appendices

- Term Language
- Normal Form
- Rules — Schematically Illustrated
  - The close rule
  - The product rule
  - The merge rule
  - The ion rule
- Proof Method
- Term Matching
  - Augmenting Rules
  - Normal Inferences
Inductive Language

Details

**Definition (Inductive language for binding bigraphs)**

The smallest set of bigraphs built by three operators —

- composition,
- tensor product \( \otimes \) (juxtaposition),
- and abstraction \( (Y)P \) of names \( Y \) on primes \( P \);

from the identities on all interfaces and a set of elementary bigraphs.

*Details on elementary bigraphs and abstraction — next slides.*
Elementary Bigraphs

Placings

Definition (2 kinds of placing)

Merge

\[ \text{merge}_n : \langle n, \emptyset, \emptyset \rangle \rightarrow \langle 1, [\emptyset], \emptyset \rangle \]

Permutation \( \pi \)

\[ \pi_{\vec{X}} : \langle m, \vec{X}, X \rangle \rightarrow \langle m, \pi(\vec{X}), X \rangle \]

(Examples) Merge and permutation

Merge

\[ \text{merge}_3 = \]

Barren root 1

\[ \text{merge}_0 = \]

Permutation \( \pi \)

\[ \{0 \mapsto 2, 1 \mapsto 0, 2 \mapsto 1\} \]

\[ \{x\}, \emptyset, \{y\} = \]

\[ \{0 \mapsto 2, 1 \mapsto 0, 2 \mapsto 1\} \]
## Term Constants

### Linkings

**Definition (2 basic kinds of linkings)**

- **Substitution** $\sigma$  
  \[ \bar{y}/\bar{X} : \langle 0, [], X \rangle \rightarrow \langle 0, [], Y \rangle \]
- **Closure**  
  \[ /X : \langle 0, [], X \rangle \rightarrow \text{id} \]

### (Examples) Linkings

- **Plain substitution** $\sigma$  
  \[ [y_1, y_2, y_3]/[\{x_1, x_2\}, \{\}, \{x_3\}] = \]

- **Renaming** $\alpha$  
  \[ [y_1, y_2, y_3]/[x_1, x_2, x_3] = \]

- **Closure**  
  \[ /\{x_1, x_2, x_3\} = \]

- **Wiring** $\omega$  
  \[ (\text{id}_{\{y_1, y_2\}} \otimes /\{z_1, z_2\}) \]
  \[ [y_1, z_1, y_2, z_2]/[\{\}, \{x_1, x_2\}, \{x_4, x_5\}, \{x_6\}] = \]

---

**Troels C. Damgaard (ITU)**  
**Syntactic Theory for Bigraphs**  
**DI, Pisa, 21-5-2007**
Definition (Concretions and Ions)

Concretion
\[ \{X\} : \langle 1, [X], X \rangle \rightarrow \langle 1, [\emptyset], X \rangle \]

Ion
\[ K_{y(x)} : \langle 1, [X], X \rangle \rightarrow \langle 1, [\emptyset], Y \rangle \]

(Examples) Concretions and ions

Concretion
\[ \{x_1, x_2\} = \]

Ion
\[ K_{y_1y_2}([\{x_1\}, \{x_2, x_3\}, \emptyset]) = \]
(Standard categorical composition and tensor product as defined earlier.)

**Definition (Abstraction)**

\[
(Y)P : I \rightarrow (1, [Y], Z \uplus Y)
\]

**Example (Abstraction)**

\[
(\{y_1, y_2\})(\{y_3\}) \setminus \{y_1, y_2, y_3, z\} = 0
\]

\[
\begin{array}{c}
\gamma_1 \gamma_2 \gamma_3 \\
0
\end{array}
\]

\[
\begin{array}{c}
\gamma_1 \gamma_2 \gamma_3 \\
Y_1 Y_2 Y_3 Z
\end{array}
\]
Bigraph $A$ — a model of an office

$$A = \begin{array}{c}
\text{secret} \\
\text{server}
\end{array} \rightarrow
\begin{array}{c}
\text{pc} \\
\text{pda} \\
\text{pda}
\end{array} \quad \text{office}$$
Normal Form

The name-discrete variant – for the equational theory

We define and prove that four forms of binding bigraph expressions generate all binding bigraphs, and that those forms are unique up to certain specified isomorphisms.

Definition (Binding discrete normal form (BDNF))

<table>
<thead>
<tr>
<th>Normal Form</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDNF</td>
<td>$M ::= (K_{\bar{y}}(\bar{x}) \otimes \text{id}_Z)P$</td>
</tr>
<tr>
<td>PDNF</td>
<td>$P ::= (Y^B) (\text{merge}_{n+k} \otimes \text{id}_Y) \left( (\bigotimes_i^n \alpha_i \downarrow) \otimes \bigotimes_i^k M_i \right) \pi$</td>
</tr>
<tr>
<td>DDNF</td>
<td>$D ::= (P_0 \otimes \ldots \otimes P_{n-1}) \pi \otimes \alpha$</td>
</tr>
<tr>
<td>BDNF</td>
<td>$B ::= (\omega \otimes \bigotimes_i^n \hat{\sigma}_i) D$</td>
</tr>
</tbody>
</table>

More details for each form on the next pages.
Free discrete molecules — MDNF

Any free discrete molecule $M : I \rightarrow \langle 1, \{\overrightarrow{y}\} \uplus Z \rangle$ can be expressed as

$$(K_{\overrightarrow{y}(\overrightarrow{x})} \otimes \text{id}_Z)P$$

where $P : I \rightarrow \langle 1, (\{\overrightarrow{X}\}), \{\overrightarrow{X}\} \uplus Z \rangle$ is a name-discrete prime.

This expression is unique up to renaming of the local names on the innerface of the ion, and (correspondingly) on the outer face of the prime $P$.

(For formal details see [Damgaard, Birkedal, ’06].)
Normal Form (cont.)

Name-discrete primes — PDNF

Any name-discrete prime \( P : I \rightarrow \langle 1, Y^B, Y \rangle \) can be expressed as

\[
(Y^B) \left( \text{merge}_{n+k} \otimes \text{id}_Y \right) \left( \Gamma \alpha_0 \otimes \cdots \otimes \Gamma \alpha_{n-1} \otimes M_0 \otimes \cdots \otimes M_{k-1} \right) \pi
\]

where every \( M_i :\rightarrow \langle Y_i^M \rangle \) is a free discrete molecule, and for renamings \( \alpha_i :\rightarrow Y_i^C \), we have \( Y = (\biguplus_{i\in n} Y_i^C) \uplus (\biguplus_{i\in k} Y_i^M) \).

This expression is unique up to reordering of the concretions and molecules, and the ordering of the sites inside the molecules; the permutation \( \pi \) changes accordingly to preserve the innerface.

(For formal details see [Damgaard, Birkedal, ’06].)
Any name-discrete bigraph $D$ with outer width $n$ can be expressed as

$$(P_0 \otimes \ldots \otimes P_{n-1})\pi \otimes \alpha$$

where every $P_i$ is a name-discrete prime, $\alpha$ is a renaming, and $\pi$ is a permutation.

This expression is unique up to reordering of the sites inside the primes; the permutation $\pi$ changes accordingly to preserve the innerface.

(For formal details see [Damgaard, Birkedal, ’06].)
Any bigraph \( G : I \rightarrow \langle n, \vec{Y}^B, \{\vec{Y}^B\} \cup Y^F \rangle \) can be expressed as

\[
\left( \omega \otimes \bigotimes_i \hat{\sigma}_i \right) D
\]

where \( D : I \rightarrow \langle n, \vec{X}, \{\vec{X}\} \cup Z \rangle \) is name-discrete, \( \omega : Z \rightarrow Y^F \) is a (global) wiring, and each \( \hat{\sigma}_i : (\vec{X}_i) \rightarrow (\vec{Y}_i^B) \) is a local substitution on the bound names of \( D \).

The expression is unique up to (local and global) renamings on the innerface of the wiring and (correspondingly) on the outerface of \( D \).

(For formal details see [Damgaard, Birkedal, ’06].)
Rules — Illustrated

This and the following sections contains illustrated versions and examples of the rules.
If \( \sigma_a, \sigma_R, \text{id}_{Y_R} \otimes \sigma_C \vdash a, R \leftrightarrow C, d \) then \( \sigma_C : \rightarrow Y \sqcup Y_C \) and \( \sigma_R : \rightarrow U \sqcup Y_R \).

\[
\frac{\sigma_a, \sigma_R, \text{id}_{Y_R} \otimes \sigma_C \vdash a, R \leftrightarrow C, d}{(\text{id} \otimes / (Y_R \sqcup Y_C)) \sigma_a, (\text{id} \otimes / Y_R) \sigma_R, (\text{id} \otimes / Y_C) \sigma_C \vdash a, R \leftrightarrow C, d}
\]
\[
\sigma_a, \sigma_R, \text{id}_{Y_R} \otimes \sigma_C \vdash a, R \leftrightarrow C, d \\
(\text{id} \otimes (Y_R \sqcup Y_C))\sigma_a, (\text{id} \otimes /Y_R)\sigma_R, (\text{id} \otimes /Y_C)\sigma_C \vdash a, R \leftrightarrow C, d
\]
Premise

\[
\frac{\omega_a, \omega_R, \omega_C \vdash a, \bigotimes^m P_{\pi^{-1}(i)} \leftrightarrow C, (\overline{\pi} \otimes id)d}{\omega_a, \omega_R, \omega_C \vdash a, \bigotimes^m P_i \leftrightarrow C\pi, d}
\]

If \( \rho = \pi^{-1} \)

\( i \)

\( a \)

\( \overline{\pi} \otimes id \)

\( d \)
\[
\begin{align*}
\text{PERM} & : \omega_a, \omega_R, \omega_C \vdash a, \bigotimes_i^m P_{\pi^{-1}(i)} \leftrightarrow C, (\bar{\pi} \otimes \text{id})d \\
& \Rightarrow \omega_a, \omega_R, \omega_C \vdash a, \bigotimes_i^m P_i \leftrightarrow \pi C, d
\end{align*}
\]
Rules — Details (cont.)

Premise

\[
\begin{align*}
\text{PRODUCT} & \quad \omega_a, \omega_R, \omega_C \parallel \omega \vdash a, R \leftrightarrow C, d \\
& \quad \omega_b, \omega_S, \omega_D \parallel \omega \vdash b, S \leftrightarrow D, e \\
& \quad \omega_a \parallel \omega_b, \omega_R \parallel \omega_S, \omega_C \parallel \omega_D \parallel \omega \vdash a \otimes b, R \otimes S \leftrightarrow C \otimes D, d \otimes e
\end{align*}
\]

If

\[
\begin{align*}
\text{id} & \quad \text{id} \\
\text{w_a} & \quad \text{id} \quad \text{w} \quad \text{w_C} \\
& \quad \text{C} \quad \text{w_R} \quad \text{id_Zd} \\
& \quad \text{id} \quad \text{id} \\
\text{a} & \quad \text{id} \quad \text{w_b} \\
& \quad \text{w_S} \quad \text{id_Zd} \\
& \quad \text{id} \quad \text{id} \\
\text{b} & \quad \text{id} \quad \text{w_D} \\
& \quad \text{id_Ze} \quad \text{e}
\end{align*}
\]
Then

\[
\omega_a, \omega_R, \omega_C \parallel \omega \vdash a, R \leftrightarrow C, d \\
\omega_b, \omega_S, \omega_D \parallel \omega \vdash b, S \leftrightarrow D, e
\]

\[
\omega_a \parallel \omega_b, \omega_R \parallel \omega_S, \omega_C \parallel \omega_D \parallel \omega \vdash a \otimes b, R \otimes S \leftrightarrow C \otimes D, d \otimes e
\]
Rules — Details (cont.)

Premise

\[
\begin{align*}
\text{ABSTR} & \quad \frac{\sigma_a \otimes \omega_a, \omega_R, \sigma_C \otimes \omega_C \vdash p, R \leftrightarrow P, d \quad \sigma_a : Z \rightarrow W \quad \sigma_C : U \rightarrow W}{\omega_a, \omega_R, \omega_C \vdash (\hat{\sigma}_a \otimes \text{id})(Z)p, R \leftrightarrow (\hat{\sigma}_C \otimes \text{id})(U)P, d}
\end{align*}
\]

If

\[
\begin{align*}
\text{id} & \quad \text{s_a} & \quad \text{w_a} \\
\text{id} & \quad \text{s_C} & \quad \text{w_C} \\
\text{id_Z} & \quad \text{w_R} & \quad \text{R} & \quad \text{d}
\end{align*}
\]
ABSTR \[ \sigma_a \otimes \omega_a, \omega_R, \sigma_C \otimes \omega_C \vdash p, R \leftrightarrow P, d \quad \sigma_a : Z \rightarrow W \quad \sigma_C : U \rightarrow W \]

\[ \omega_a, \omega_R, \omega_C \vdash (\hat{\sigma_a} \otimes \text{id})(Z)p, R \leftrightarrow (\hat{\sigma_C} \otimes \text{id})(U)P, d \]

Then

\[ (Z)p \]

\[ \omega_a \]

\[ \sigma_a \]

\[ \omega_C \]

\[ \sigma_C \]

\[ \text{id} \]

\[ (s_a) \]

\[ (s_C) \]

\[ (U)P \]

\[ R \]

\[ d \]
Rules — Details (cont.)

Premise

\[ \omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d \quad \text{aglobal} \]

\[ \omega_a, \omega_R, \omega_C \vdash (\text{merge} \otimes \text{id})a, R \leftrightarrow (\text{merge} \otimes \text{id})C, d \]

If

\[
\begin{array}{c}
\text{id} \\
\text{w}_a \\
\text{w}_c \\
\text{id}_Z
\end{array}
\]

= 

\[
\begin{array}{c}
\text{id} \\
\text{C} \\
\text{w}_R \\
\text{R} \\
\text{d}
\end{array}
\]
Rules — Details (cont.)

Conclusion

\[ \frac{\omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d}{\omega_a, \omega_R, \omega_C \vdash (\text{merge} \otimes \text{id})a, R \leftrightarrow (\text{merge} \otimes \text{id})C, d} \]

\[ \text{MERGE} \]

Then

\[ \frac{\omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d}{\frac{\omega_a, \omega_R, \omega_C \vdash (\text{merge} \otimes \text{id})a, R \leftrightarrow (\text{merge} \otimes \text{id})C, d}{\frac{\text{id} \quad \omega_a}{\frac{\text{id}}{\omega_c}}}{\frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}}}{\frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]

\[ \frac{\text{id} \quad \text{id}}{\frac{\text{id}}{\text{merge} \quad \text{id}}}{\frac{\text{merge} \quad \text{id}}{\frac{\text{id} \quad \omega_c}{\frac{\text{id}}{\omega_R}}}} \]
Premise

ION  \[ \omega_a, \omega_R, \omega_C \vdash ((\vec{v})/\vec{X}) \otimes \text{id} p, R \hookrightarrow ((\vec{v})/\vec{Z}) \otimes \text{id} P, d \quad \alpha = \vec{y}/\vec{u} \quad \sigma : \{\vec{y}\} \rightarrow \]

\[ \sigma \parallel \omega_a, \omega_R, \sigma\alpha \parallel \omega_C \vdash (K_{\vec{y}(\vec{X})} \otimes \text{id}) p, R \hookrightarrow (K_{\vec{u}(\vec{Z})} \otimes \text{id}) P, d \]

If

(\text{here } (s_X) = (\vec{v})/\vec{X}) \text{ and } (s_Z) = (\vec{v})/\vec{Z}).
Conclusion

\[
\text{ION} \quad \omega_a, \omega_R, \omega_C \vdash ((\vec{v})/(\vec{X}) \otimes \text{id})p, R \leftrightarrow ((\vec{v})/(\vec{Z}) \otimes \text{id})P, d \quad \alpha = \vec{y}/\vec{u} \quad \sigma : \{\vec{y}\} \rightarrow
\]

\[
\sigma \parallel \omega_a, \omega_R, \sigma \alpha \parallel \omega_C \vdash (K_{\vec{y}(\vec{X})} \otimes \text{id})p, R \leftrightarrow (K_{\vec{u}(\vec{Z})} \otimes \text{id})P, d
\]
$\omega_a, id_e, \omega_C(\sigma \otimes \omega_R \otimes id) \vdash p, id \leftrightarrow P, d$

$P :\rightarrow \langle W \cup Y \rangle$

$\sigma : W \rightarrow U$

$\omega_a, \omega_R, \omega_C \vdash p, (\hat{\sigma} \otimes id)(W)P \leftrightarrow \{ U \}, d$

If

\begin{align*}
\text{id} & \quad \text{id} \quad \text{w}_a \\
\text{id} & \quad \text{id} \quad \text{w}_C
\end{align*}

= 

\begin{align*}
\text{id} & \quad \text{id} \quad \text{s} \quad \text{w}_R \\
\text{id} & \quad \text{id} \quad \text{id} \quad \text{id}_Z \\
\text{P} & \quad \text{id} \\
\text{d}
\end{align*}
\[
\begin{align*}
\text{SWITCH } \omega_a, \text{id}_e, \omega_C (\sigma \otimes \omega_R \otimes \text{id}) & \vdash p, \text{id} \leftrightarrow P, d \quad P : \to \langle W \cup Y \rangle \\
\omega_a, \omega_R, \omega_C & \vdash p, (\hat{\sigma} \otimes \text{id})(W)P \leftrightarrow \Gamma U, d
\end{align*}
\]
The **CLOSE** rule illustrated

\[
\begin{aligned}
\sigma_a, \sigma_R, \text{id}_{Y_R} \otimes \sigma_C & \vdash a, R \leftrightarrow C, d \\
\sigma_C & \rightarrow Y \uplus Y_C \\
\sigma_R & \rightarrow U \uplus Y_R \\
\text{id} \otimes (Y_R \uplus Y_C) & \sigma_a, (\text{id} \otimes Y_R) \sigma_R, (\text{id} \otimes Y_C) \sigma_C \vdash a, R \leftrightarrow C, d
\end{aligned}
\]
The **PRODUCT** rule illustrated (cont.)

![Diagram of the PRODUCT rule](image-url)
The **PRODUCT** rule illustrated

\[ \begin{align*}
\omega_a & \vdash a, R \leftrightarrow C, d \\
\omega_b & \vdash b, S \leftrightarrow D, e
\end{align*} \]

\[ \begin{align*}
\omega_a, \omega_R, \omega_C & \parallel \omega \vdash a \otimes b, R \otimes S \leftrightarrow C \otimes D, d \otimes e
\end{align*} \]
Rules — Illustrated (cont.)

MERGE

\[
\begin{align*}
\text{id} & \quad \text{id} \\
\text{vw}_a & \quad \text{vw}_c \\
\hline
\text{a} & \quad \text{C} \\
\text{R} & \quad \text{id}_Z \\
\text{d} & \\
\end{align*}
\]

\[
\begin{align*}
\text{id} & \quad \text{id} \\
\text{vw}_a & \quad \text{vw}_c \\
\hline
\text{merge} & \quad \text{id} \\
\text{id} & \quad \text{merge} \\
\text{a} & \quad \text{C} \\
\text{R} & \quad \text{id}_Z \\
\text{d} & \\
\end{align*}
\]
The ION rule illustrated

\[
\begin{align*}
\alpha' & \quad \text{a'} \\
y_1' & \quad \text{y}_1' \\
y_2' & \quad \text{y}_2' \\
y_1 & \quad \text{y}_1 \\
y_2 & \quad \text{y}_2 \\
K & \quad \text{K} \\
\end{align*}
\]

\[
\begin{align*}
\alpha & \quad \text{a} \\
y_1' & \quad \sigma_y \text{y}_1' \\
y_2' & \quad \sigma_y \text{y}_2' \\
y_1 & \quad \sigma_y \text{y}_1 \\
y_2 & \quad \sigma_y \text{y}_2 \\
K & \quad \text{K} \\
\end{align*}
\]

\[
\begin{align*}
\beta & \quad \text{C'} \\
y_1' & \quad \sigma_y \alpha \text{y}_1' \\
y_2' & \quad \sigma_y \alpha \text{y}_2' \\
y_1 & \quad \sigma_y \alpha \text{y}_1 \\
y_2 & \quad \sigma_y \alpha \text{y}_2 \\
K & \quad \text{K} \\
\end{align*}
\]

\[
\begin{align*}
\omega_a, \omega_R, \omega_C \vdash (\overline{v}/(\overline{X}) \otimes \text{id})p, R \leftrightarrow ((\overline{v}/(\overline{Z}) \otimes \text{id})p, d, \quad \alpha = \overline{y}/\overline{u} \quad \sigma : \{\overline{y}\} \rightarrow \\
\sigma \parallel \omega_a, \omega_R, \sigma\alpha \parallel \omega_C \vdash (K\overline{y}(\overline{x}) \otimes \text{id})p, R \leftrightarrow (K\overline{u}(\overline{z}) \otimes \text{id})p, d
\end{align*}
\]
Proof Method (Completeness)

Definition

The size of a matching sentence \( \omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d \) is the number of ions in \( a \).

Proof method

- Establish a series of lemmas that express how a valid sentence may be derived by applications of inference rules to valid sentences of lesser or equal size.
- Use normal form theorems to help
  - decompose components of given valid sentence; and,
  - verify that the claimed valid sentences, do exist.
- in particular, unicity results for normal form theorems yield a number of equalities for decomposed parts, which help here.

Lemmas and main theorem on following pages.
Proof Method (Completeness) (cont.)

Lemma (1)

Every valid sentence $\omega_a, \omega_R, \omega_C \models a, R \leftrightarrow C, d$ is provable using the \textit{CLOSE} and the \textit{PERM} rule on a valid sentence, of equal size, of the form $\sigma_a, \sigma_R, \sigma_C \models a, S \leftrightarrow \bigotimes_i^n P_i, e$.

Lemma (2)

Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models a, R \leftrightarrow P \otimes \bigotimes_i^n P_i, d$, with $P$ and $P_i$ prime and discrete, is provable using the \textit{PRODUCT} rule on valid sentences, of lesser or equal size, of the form $\sigma^P_a, \sigma^P_R, \sigma^P_C \parallel \sigma^S_C \models p, S \leftrightarrow P, e$ and $\sigma^C_a, \sigma^C_R, \sigma^C_C \parallel \sigma^S_C \models a', R' \leftrightarrow \bigotimes_i^n P_i, e'$.
Proof Method (Completeness) (cont.)

Lemma (3)

Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models a, R \leftrightarrow \text{id}_d, d$ is provable using PRODUCT and WIRING-AXIOM.

Lemma (4)

Every valid sentence $\omega_a, \omega_R, \omega_C \models p, R \leftrightarrow P, d$, with $p$ and $P$ prime and discrete, is provable using the ABSTR rule on a valid sentence, of lesser or equal size, of the form $\omega'_a, \omega'_R, \omega'_C \models p', R \leftrightarrow P', d$, where $p'$ and $P'$ are discrete free primes.
Proof Method (Completeness) (cont.)

Lemma (5)

Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models p, R \leftrightarrow P, d$, with $p$ and $P$ discrete and free primes, is provable using MERGE, PRODUCT (iterated), and SWITCH rules on valid sentences, each of lesser or equal size, and each on one of two forms:

- $\sigma_a', \sigma_R', \sigma_C' \models p^N, \text{id} \leftrightarrow P^N, e$, where $p^n$ and $P^N$ are free discrete primes,
- $\sigma_a', \sigma_R', \sigma_C' \models m, S \leftrightarrow M, e$, where $m$ and $M$ are free discrete molecules.

Lemma (6)

Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models m, R \leftrightarrow M, d$, with $m$ and $M$ free discrete molecules, is provable using the ION rule on a valid sentence $\sigma_a', \sigma_R', \sigma_C' \models p, R \leftrightarrow P, d$, of lesser size, where $p$ and $P$ are discrete primes.

Lemma (7)

Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models p, \text{id} \leftrightarrow P, e$, with $p$ and $P$ free discrete primes, is provable using the MERGE and PRODUCT (iterated) rules on valid sentences of equal or lesser size, which are either instances of rule PAX or of the form $\sigma_a', \sigma_r', \sigma_M' \models m, R \leftrightarrow M, d$. 

Theorem

The rules for matching are complete, that is, any valid matching sentence can be derived from the rules.

Proof.

By induction on the size of a sentence. By the lemmas above, we have that all valid sentences with size $n$ can be derived from valid sentences of the form $\sigma_a, \sigma_R, \sigma_C \models m, R \leftrightarrow M, d$, with $m$ and $M$ free discrete molecules, of size less than or equal to $n$. By Lemma 6, these can be derived from sentences of size less than $n$. 
Reformulating **PRODUCT** for Bigraph Terms

—separating associative grouping from subterm matching

\[
\begin{align*}
\text{PAR} & \quad \frac{\omega_a, \omega_R, \omega_C \parallel \omega \vdash a, R \hookrightarrow C, d}{\omega_a \parallel \omega_b, \omega_R \parallel \omega_S, \omega_C \parallel \omega_D \parallel \omega \vdash a \otimes b, R \otimes S \hookrightarrow C \otimes D, d \otimes e}
\end{align*}
\]

- The binary PAR rule is replaced by
  - an iterative rule taking \(n\) agent and redex parts, and
  - an equivalence rule grouping redex primes to match agent primes

**PAR rules for bigraph terms**

\[
\begin{align*}
\text{PAR}_n & \quad \frac{\sigma_I : I_R \rightarrow I_a \quad \sigma^a_i, \sigma^R_i, \sigma \parallel \sigma^C_i \vdash e_i, \bar{P}_i \leadsto E_i, \bar{q}_i}{(I_a \parallel \|_i^n \sigma^a_i), (I_R \parallel \|_i^n \sigma^R_i), (\sigma_I \parallel \sigma \parallel \|_i^n \sigma^C_i) \vdash \boxtimes_i^n e_i, \boxtimes_i^n \bar{P}_i \leadsto \boxtimes_i^n E_i, \boxtimes_i^n \bar{q}_i}
\end{align*}
\]

\[
\begin{align*}
P'_{ij} & \equiv P_j + \sum_{r \in i} l_r \quad q'_{ij} \equiv q_j + \sum_{r \in i} k_r \quad P'_{ij} : \langle k_{ij}, \tilde{X}_{ij} \rangle \rightarrow \quad k_i = \sum_{j \in l_i} k_{ij} \\
& \quad \frac{\sigma^a, \sigma^R, \sigma^C \vdash \boxtimes_i^n e_i, \boxtimes_i^n \otimes_j^l P'_{ij} \leadsto \boxtimes_i^n E_i, \boxtimes_i^n \otimes_j^k q'_{ij}}{\sigma^a, \sigma^R, \sigma^C \vdash \boxtimes_i^n e_i, \boxtimes_i^m P_i \leadsto \boxtimes_i^n E_i, \boxtimes_i^{m'} q_i}
\end{align*}
\]
Reformulating **MERGE** for Bigraph Terms
—making molecule matching explicit

### MERGE rule for bigraph terms

\[
\omega_a, \omega_R, \omega_C \vdash a, R \mapsto C, d \quad a \text{ global}
\]

\[
\omega_a, \omega_R, \omega_C \vdash (merge \otimes id)a, R \mapsto (merge \otimes id)C, d
\]

- The MERGE rule is replaced by a rule that makes explicit how agent molecules should be partitioned into sets.
- Each redex prime is matched within one set of molecules.

\[
\rho \in \bar{\rho}(n, m)
\]

\[
\sigma^a, \sigma^R, \sigma^C \vdash \bigotimes_i^m (id \otimes merge) \bigotimes_{j \in \varrho_i} m_j, \bar{P} \leadsto (\bigotimes_i^m S_{\pi(i)})\bar{\pi}, \bar{q}
\]

\[
\sigma^a, \sigma^R, \sigma^C \vdash (id \otimes merge) \bigotimes_i^n m_i, \bar{P} \leadsto (id \otimes merge) \bigotimes_i^m S_i, \bar{q}
\]

\((\bar{\rho}(n, m)\) is the set of partitionings of 0, \ldots, \(n - 1\) into \(m\) sets.)
We get the following inference system:

\[
\begin{array}{llllll}
\text{PAX} & \sigma : W \sqcup Z \rightarrow \alpha : V \rightarrow W & \tau : X \rightarrow V & g : \langle X \sqcup Z \rangle \\
\frac{}{\sigma (\text{id}_Z \otimes \alpha \tau), \text{id}_V, \sigma \vdash g, \text{id}_V \leadsto \alpha \sqcup (\text{id}_Z \otimes \hat{\tau})(X)g}
\end{array}
\]

\[
\begin{array}{llllll}
\text{ABS} & \sigma^a \otimes \sigma^a, \sigma^R, \sigma^C \sqcap g, \overline{P} \leadsto G, \overline{q} & \sigma^a : Z \rightarrow W & \sigma^a : U \rightarrow W \\
\frac{}{\sigma^a, \sigma^R, \sigma^C \vdash (\text{id} \otimes \sigma^a_L)(Z)g, \overline{P} \leadsto (\text{id} \otimes \sigma^a_L)(U)G, \overline{q}}
\end{array}
\]

\[
\begin{array}{llllll}
\text{ION} & \sigma^a, \sigma^R, \sigma^C \vdash (\text{id} \otimes (\overline{v})/\overline{X})n, \overline{P} \leadsto (\text{id} \otimes (\overline{v})/\overline{Z})N, \overline{q} & \alpha = \overline{y}/\overline{u} & \sigma : \{\overline{y}\} \rightarrow \\
\frac{}{(\sigma \parallel \sigma^a), \sigma^R, (\sigma \alpha \parallel \sigma^C) \vdash (\text{id} \otimes K_{\overline{y}(\overline{x})})n, \overline{P} \leadsto (\text{id} \otimes K_{\overline{u}(\overline{z})})N, \overline{q}}
\end{array}
\]

\[
\begin{array}{llllll}
\text{SWX} & \sigma^a, \text{id}_V, \sigma^C(\text{id} \otimes \sigma \otimes \sigma^R) \vdash g, \bigotimes_i^n \text{id} \leadsto G, \overline{q} & G : \rightarrow \langle W \sqcup Z \rangle & \sigma : W \rightarrow U \\
\frac{}{\sigma^a, \sigma^R, \sigma^C \vdash g, (\text{id} \otimes \hat{\sigma})(W)G \leadsto \hat{U} \sqcup, \overline{q}}
\end{array}
\]
Inference System for Bigraph Term Matching (cont.)

\[
\text{PAR}_n: \quad \sigma_I : I_R \rightarrow I_a \quad (\forall i \in n) \quad \sigma_i^a, \sigma_i^R, \sigma \parallel \sigma_i^C \vdash e_i, \bar{P}_i \sim E_i, \bar{q}_i
\]

\[
(I_a \parallel \bigotimes_i^n \sigma_i^a), (I_R \parallel \bigotimes_i^n \sigma_i^R), (\sigma_I \parallel \sigma \parallel \bigotimes_i^n \sigma_i^C) \vdash \bigotimes_i^n e_i, \bigotimes_i^n \bar{P}_i \sim \bigotimes_i^n E_i, \bigotimes_i^n \bar{q}_i
\]

\[
P_{ij}' = P_j + \sum_{r \in l} l_r \quad q_{ij}' = q_j + \sum_{r \in k} k_r \quad P_{ij}' : \langle k_{ij}, \bar{X}_{ij} \rangle \rightarrow \quad k_i = \sum_{j \in l_i} k_{ij}
\]

\[
\text{PAR}_\equiv: \quad \sigma^a, \sigma^R, \sigma^C \vdash \bigotimes_i^n e_i, \bigotimes_i^n \bigotimes_j l_i P_{ij} \sim \bigotimes_i^n E_i, \bigotimes_i^n q_{ij}'
\]

\[
\sigma^a, \sigma^R, \sigma^C \vdash \bar{e}, \bigotimes_i^n Q_{\pi - 1}(i) \sim \bar{E}, \bigotimes_i^m q_{\pi - 1}(i)
\]

\[
\text{PER}: \quad \rho \in \bar{q}(n, m) \quad \sigma^a, \sigma^R, \sigma^C \vdash \bigotimes_i^m (id \otimes \text{merge}) \bigotimes_{j \in \varrho_i} m_j, \bar{P} \sim (\bigotimes_i^m S_{\pi(i)}) \bar{\pi}, \bar{q}
\]

\[
\sigma^a, \sigma^R, \sigma^C \vdash (id \otimes \text{merge}) \bigotimes_i^m m_i, \bar{P} \sim (id \otimes \text{merge}) \bigotimes_i^m S_i, \bar{q}
\]

\[
\text{MER}: \quad \sigma^a, \sigma^R, \sigma^C \vdash \bigotimes_i^m (id \otimes \text{merge}) \bigotimes_{j \in \varrho_i} m_j, \bar{P} \sim (id \otimes \text{merge}) \bigotimes_i^m S_{\pi(i)} \bar{\pi}, \bar{q}
\]

\[
\sigma^a, \sigma^R, id_{Y_R} \otimes \sigma^C \vdash \bar{p}, \bar{P} \sim \bar{Q}_{\pi}, \bar{q}
\]

\[
(id \otimes / (Y_R \cup Y_C)) \sigma^a, (id \otimes / Y_R) \sigma^R, (id \otimes / Y_C) \sigma^C \vdash \bar{p}, \bar{P} \sim \bar{Q}_{\pi}, \bar{q}
\]

(Plus normalizing-rule DNF.)
Normal Inference Grammar
—restricts the order of rules in inference trees

- We let a **normal inference** be defined by $\mathcal{D}_N$ in this grammar:

<table>
<thead>
<tr>
<th>Normal inference tree</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{D}_g' := \begin{cases} \text{PAX} \ldots \ \text{PAR}_n^G \mathcal{D}_g' \ldots \mathcal{D}_g' \ \text{PAR}_G^G \ldots \ \text{PER}_G^G \ldots \ \text{MER} \ldots \ \text{ION} \mathcal{D}_p' \ \ldots \end{cases}$</td>
<td>$\mathcal{D}_g := \begin{cases} \text{SWX} \ldots \ \text{PAX} \ldots \ \text{PAR}_n^G \mathcal{D}_g \ldots \mathcal{D}_g \ \text{PAR}_G^G \ldots \ \text{PER}_G^G \ldots \ \text{MER} \ldots \ \text{ION} \mathcal{D}_p \ \ldots \end{cases}$</td>
<td>$\mathcal{D}_p := \begin{cases} \text{ABS} \mathcal{D}_g' \ \ldots \end{cases}$</td>
</tr>
<tr>
<td>$\mathcal{D}_p' := \text{ABS} \mathcal{D}_g'$</td>
<td>$\mathcal{D}_p := \text{ABS} \mathcal{D}_g$</td>
<td>$\mathcal{D}_N := \text{CLO}$</td>
</tr>
</tbody>
</table>

- This has been proven to still be complete.
Where is the Nondeterminism Hidden?

There are several sources of nondeterminism:

- Given normal inference, choice of rule is limited to $D_G$ and $D'_G$
- Grouping (parenthesisation) of tensor product in $\text{PAR}_\cong$ rule

\[
\text{PAR}_\cong : \quad \ldots \vdash \bigotimes^n_i e_i, \bigotimes^n_i \bigotimes^l_i P_{ij} \leadsto \bigotimes^n_i E_i, \bigotimes^n_i \bigotimes^k_i q'_{ij}
\]

\[
\ldots \vdash \bigotimes^n_i e_i, \bigotimes^m_i P_i \leadsto \bigotimes^n_i E_i, \bigotimes^m_i q_i
\]

- Partitioning of molecules by $\varrho$ in $\text{MER}$ rule

\[
\text{MER} : \quad \varrho \in \overline{\varrho}(n, m) \quad \ldots \vdash \bigotimes^m_i (\text{id} \otimes \text{merge}) \bigotimes_{j \in \varrho_i} m_j, \overline{P} \leadsto \ldots
\]

\[
\ldots \vdash (\text{id} \otimes \text{merge}) \bigotimes^n_i m_i, \overline{P} \leadsto \ldots
\]

- Permutation of redex primes in $\text{PER}$ rule

\[
\text{PER} : \quad \ldots \vdash \overline{e}, \bigotimes^n_i Q_{\pi^{-1}(i)} \leadsto \overline{E}, \bigotimes^m_i q_{\pi^{-1}(i)}
\]

\[
\ldots \vdash \overline{e}, \bigotimes^n_i Q_i \leadsto \overline{E\pi}, \bigotimes^m_i q_i
\]

- Choice of $\sigma, \alpha, \tau$ in $\text{PAX}$

\[
\text{PAX} : \quad \sigma : W \uplus Z \rightarrow \alpha : V \rightarrow W \quad \tau : X \rightarrow V \quad g : \langle X \uplus Z \rangle
\]

\[
\frac{\sigma \left( \text{id}_Z \otimes \alpha \tau \right), \text{id}_\epsilon, \sigma \vdash g, \text{id}_{(V)} \leadsto \neg \alpha \neg^{\uplus}, (\text{id}_Z \otimes \hat{\tau})(X)g}{\sigma, \alpha, \tau}
\]