SYNTACTIC THEORY FOR BIGRAPHS

Troels C. Damgaard (tcd@itu.dk)

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

Bigraphical Programming Languages (BPL) Group
IT University of Copenhagen

4 May 2007 / École Normale Supérieure, Paris
RELEVANT PUBLICATIONS


See my webpage for more information (e.g., links to more publications and to the BPL group):

www.itu.dk/people/tcd/
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
   - Characterization
   - Rules — Details
6 Current and Future Work
7 Appendix (Gritty Details)
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
   • Characterization
   • Rules — Details

6 CURRENT AND FUTURE WORK

7 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 theory for reactive systems, where many existing calculi for mobility and concurrency may be represented and investigated.

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.
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 bigraph model of an office

\[ A = \]

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 $\otimes$. 

$$B : \langle 2, (\{x\}, \emptyset), \{x, z\} \rangle \to \langle 2, (\emptyset, \emptyset), \emptyset \rangle$$

$$C : \langle 0, (\), \emptyset \rangle \to \langle 2, (\{x\}, \emptyset), \{x, z\} \rangle$$
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
   - Characterization
   - Rules — Details

6. **CURRENT AND FUTURE WORK**

7. **APPENDIX (GRITTY DETAILS)**
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 $G_0 G_1$ — 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 BIGRAPHHS (CONT.)

BUILDING AN EXPRESSION FOR A

\[ \text{id}_1 \otimes \{y_0, y_1\} \otimes \text{id}_1 \otimes \{z_0, z_1\} \]
LANGUAGE FOR BIGRAPHS (CONT.)

BUILDING AN EXPRESSION FOR A

\[
\text{id}_1 \otimes \left/ \{y_0, y_1\} \otimes \text{id}_1 \otimes \left/ \{z_0, z_1\} \right.
\]

server \(y_0(x)\)

\text{server}
LANGUAGE FOR BIGRAPHS (CONT.)

BUILDING AN EXPRESSION FOR $A$

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

server $y_0(x)$

(server)

(secret $x$)

$$(x) \text{ secret}_x$$
Building an expression for $A$

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

$$\text{secret}_x$$

$$1$$

$$\text{id}_1 \otimes /\{y_0, y_1\} \otimes \text{id}_1 \otimes /\{z_0, z_1\}$$
**Language for Bigraphs (cont.)**

**Building an expression for A**

\[ \text{id}_1 \otimes \{y_0, y_1\} \otimes \text{id}_1 \otimes \{z_0, z_1\} \]

\[ y_1/y_1 \otimes \text{office} \otimes z_0/z_0 \otimes z_1/z_1 \]
LANGUAGE FOR BIGRAPHS (CONT.)

BUILDING AN EXPRESSION FOR A

server\_{y_0(x)}

(y_0, y_1) \otimes id_1 \otimes /\{z_0, z_1\}

(y_1/y_1 \otimes office \otimes z_0/z_0 \otimes z_1/z_1)

(x) secret\_x

(y_1/y_1 \otimes merge_3 \otimes z_0/z_0 \otimes z_1/z_1)

1
BUILDING AN EXPRESSION FOR A

\[
\begin{align*}
\text{server}_y(x) & \quad \text{secret}_x \\
& \quad \text{office} \\
1 & \quad \text{pc}_{y_1} \otimes \text{pda}_{z_0} \otimes \text{pda}_{z_1} \\
\end{align*}
\]

\[
\begin{align*}
\text{id}_1 \otimes \{y_0, y_1\} \otimes \text{id}_1 \otimes \{z_0, z_1\} \\
y_1/y_1 \otimes \text{office} \otimes z_0/z_0 \otimes z_1/z_1 \otimes z_1 \\
y_1/y_1 \otimes \text{merge}_3 \otimes z_0/z_0 \otimes z_1/z_1 \otimes z_1 \\
\end{align*}
\]
BUILDING AN EXPRESSION FOR $A$

Language for Bigraphs (cont.)

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

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

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

$$pc_{y_1} \otimes \text{pda}_{z_0} \otimes \text{pda}_{z_1}$$

1 $\otimes$ 1 $\otimes$ 1
The terms in the language are explicitly typed with interfaces —
determining when tensor product, composition, and abstraction is well
defined.

Hence, formation rules ensure well formed terms denote bigraphs.

The interface for expressions can be determined by induction.

Similarly, the bigraph denoted by an expression can be determined by
induction.
Theorem (Completeness of Language)

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

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

...follows immediately from the normal form theorem.
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);
- 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;

\[ B = \begin{array}{c}
\text{id} \\
\hline
w \\
\hline
D
\end{array} \]
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,
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 bigraph decomposition](image)
Main theorem states 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.*
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
   • Characterization
   • Rules — Details

6 CURRENT AND FUTURE WORK

7 APPENDIX (GRITTY DETAILS)
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:

\[(\text{pc}_{y_1} \otimes \text{pda}_{z_0}) (1 \otimes 1) = (\text{pc}_{y_1} 1 \otimes \text{pda}_{z_0} 1),\]

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

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)

\begin{align*}
(\text{C1}) & \quad A \ id_I = A = \ id_J A \\ 
(\text{C2}) & \quad A(BC) = (AB)C \\ 
(\text{C3}) & \quad A \otimes \ id_{\epsilon} = A = \ id_{\epsilon} \otimes A \\ 
(\text{C4}) & \quad A \otimes (B \otimes C) = (A \otimes B) \otimes C \\ 
(\text{C5}) & \quad \ id_I \otimes \ id_J = \ id_I \otimes J \\ 
(\text{C6}) & \quad (A_1 \otimes B_1)(A_0 \otimes B_0) = (A_1 A_0) \otimes (B_1 B_0) \\ 
(\text{C7}) & \quad \gamma_{I,\epsilon} = \ id_I \\ 
(\text{C8}) & \quad \gamma_{J,I} \gamma_{I,J} = \ id_I \otimes J \\ 
(\text{C9}) & \quad \gamma_{I \otimes J,K} = (\gamma_{I,K} \otimes \ id_J)(\ id_I \otimes \gamma_{J,K}) \\ 
(\text{C10}) & \quad \gamma_{I,K}(A \otimes B) = (B \otimes A)\gamma_{H,J} \quad (A : H \rightarrow I, B : J \rightarrow K)
\end{align*}
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 explain the basic ways in which placing expressions for equal bigraphs can vary.

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

Two axioms for ions deal with renaming of the names on both the interfaces of an ion.

\[(N1) \quad (\text{id}_1 \otimes \alpha)K_{\bar{y}(\bar{x})} = K_{\alpha(\bar{y})(\bar{x})}\]
\[(N2) \quad K_{\bar{y}(\bar{x})\hat{\sigma}} = K_{\bar{y}(\hat{\sigma}^{-1}(\bar{x}))}\]
BINDING AXIOMS

Binding axioms state basic equalities between expressions with the abstraction operator and/or the concretion constant.

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

\[(Y)\downarrow Y\downarrow = \text{id}(Y)\]
abstracting a concretion of the names \(Y\) is just an identity on \(Y\).

\[(\uparrow X\downarrow Z \otimes \text{id}_Y)(X)P = P\]
“concreting” an abstraction of the names \(X\) gives you back the bigraph \(P\) that you started with.

\[(\text{id}_X \otimes (Y)P)\text{G} = (Y)(P \otimes \text{id}_X)\text{G}\]
extending the scope of an abstraction over a set of global names \(X\) is ok, when the entire expression has a local innerface.

\[(X \uplus Y)P = (X)(Y)P\]
abstracting and abstracting again is the same as abstracting the union of the names involved.
**CONCLUSION**

**MAIN RESULT - SOUNDNESS AND COMPLETENESS OF THE EQUATIONAL THEORY**

**THEOREM (SOUNDNESS AND COMPLETENESS)**

For all binding bigraph expressions $E$ and $F$, $\llbracket E \rrbracket = \llbracket F \rrbracket$, i.e., $E$ and $F$ denote the same bigraph iff $\vdash E = F$. 
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**

So, we have definitions

**Definition (A Bigraph)**

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

**Definition (A Match in Bigraphs)**

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

**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,
- hope to be able to specialize the characterization into an algorithmic approach.
We define a new representation for a match — as a relation — and look to inductively characterize this relation.

**Definition (Matching sentence)**

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

for wirings (essentially, link graphs) \( \omega_a, \omega_R, \omega_C \) and discrete bigraphs \( a, R, C, d \).

- \( a, d \) are ground,
- \( R \) and \( C \) are products of discrete primes,
- \( R \) is can be constructed without using permutations (convenience).
**Definition (Valid matching sentence)**

A matching sentence as above is valid iff

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

Valid matching sentences recapture original definition of matching; as we have simply decomposed discretely agent, context, redex and parameter.
**Rules for Deriving Valid Matching Sentences**

**Matching rules**

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

We take two axioms,

- one concerned with introducing nonconnected linkage; and,
- one, which allows to conclude when we have roots with equal structure in agent and parameter (up to certain renamings).

And seven rules, most of which serve to introduce an elementary bigraph or operator.

For example,

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

and,

\[
\frac{\omega_a, \omega_R, \omega_C \vdash a, R \leadsto C, d \text{ a global}}{\omega_a, \omega_R, \omega_C \vdash (merge \otimes \text{id})a, R \leadsto (merge \otimes \text{id})C, d}.
\]
RULES — ILLUSTRATED (CONT.)

MERGE

id

\[ w_a \]

\[
\begin{array}{c}
\text{a} \\
\end{array}
\]

= 

id

\[ w_c \]

\[
\begin{array}{c}
\text{C} \\
\text{w_R} \\
\text{id_Z} \\
\text{R} \\
\text{d} \\
\end{array}
\]

id

\[ w_a \]

\[
\begin{array}{c}
\text{id} \\
\text{merge} \\
\text{a} \\
\text{R} \\
\text{d} \\
\end{array}
\]

= 

id

\[ w_c \]

\[
\begin{array}{c}
\text{id} \\
\text{merge} \\
\text{id} \\
\text{w_R} \\
\text{id_Z} \\
\text{C} \\
\text{R} \\
\text{d} \\
\end{array}
\]
RULES - DETAILS

**CLOSE**

\[
\begin{align*}
\sigma_a, \sigma_R, \text{id}_{Y_R} \otimes \sigma_C & \vdash a, R \leftrightarrow C, d & \sigma_C : \rightarrow Y \cup Y_C & & \sigma_R : \rightarrow U \cup Y_R \\
(\text{id} \otimes ((Y_R \cup Y_C))\sigma_a, (\text{id} \otimes Y)\sigma_R, (\text{id} \otimes Y_C)\sigma_C & \vdash a, R \leftrightarrow C, d
\end{align*}
\]

**PERM**

\[
\begin{align*}
\omega_a, \omega_R, \omega_C & \vdash a, \bigotimes_i^m P_{\pi^{-1}(i)} \leftrightarrow C, (\overline{\pi} \otimes \text{id})d \\
\omega_a, \omega_R, \omega_C & \vdash a, \bigotimes_i^m P_i \leftrightarrow C\pi, d
\end{align*}
\]

**PRODUCT**

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

**LSUB**

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

**MERGE**

\[
\begin{align*}
\omega_a, \omega_R, \omega_C & \vdash a, R \leftrightarrow C, d & \text{a global} \\
\omega_a, \omega_R, \omega_C & \vdash (\text{merge} \otimes \text{id})a, R \leftrightarrow (\text{merge} \otimes \text{id})C, d
\end{align*}
\]
\[
\omega_a, \omega_R, \omega_C \vdash ((\bar{v})/(\bar{X}) \otimes \text{id})p, R \leftrightarrow ((\bar{v})/(\bar{Z}) \otimes \text{id})P, d \quad \alpha = \bar{y}/\bar{u} \quad \sigma : \{\bar{y}\} \rightarrow \sigma \parallel \omega_a, \omega_R, \sigma \alpha \parallel \omega_C \vdash (K\bar{y}(\bar{x}) \otimes \text{id})p, R \leftrightarrow (K\bar{u}(\bar{z}) \otimes \text{id})P, d
\]

\[
\omega_a, \text{id}_\epsilon, \omega_C(\sigma \otimes \omega_R \otimes \text{id}) \vdash p, \text{id} \leftrightarrow P, d \quad P : \rightarrow \langle W \cup Y \rangle \quad \sigma : W \rightarrow U
\]

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

\[
\alpha : X \rightarrow U \quad \beta : Z \rightarrow \quad \sigma_L : Y \rightarrow U \quad p : \langle Y \cup Z \rangle \\
\sigma(\sigma_L \otimes \beta), \text{id}_\epsilon, \sigma \vdash p, \text{id}(X) \leftrightarrow \Gamma (X)(\alpha^{-1} \sigma_L \otimes \beta \otimes \text{id}_1)p
\]

\[
y, X, y/X \vdash \text{id}_\epsilon, \text{id}_\epsilon \leftrightarrow \text{id}_\epsilon, \text{id}_\epsilon
\]
**CONCLUSION**

**MAIN RESULT - SOUNDNESS AND COMPLETENESS OF THE RULES**

---

**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, any valid matching sentence can be derived from the rules.*

The equality theory and normal forms are employed heavily to prove these theorems.
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
   - Characterization
   - Rules — Details

6. Current and Future Work

7. Appendix (Gritty Details)
CURRENT WORK

TERM MATCHING

In the BPL group we build a tool to experiment with bigraphical reactive systems.

Bigraphs are represented using a term language with constants and operators corresponding exactly to the elementary bigraphs and combinators.

Based on the work presented — we use a simple approach to implementing matching in such a tool:

First step: Recast rules to work on terms — one simply has to add a single rule STRUCT to allow application of structural congruence on terms.
NORMAL INFERENCE VS. ALGORITHMIC APPROACH

Each definition of normal inferences correspond to a different algorithmic approach to matching.

Might even be worthwhile extending the set of rules to allow even more freedom in choosing a definition of normal inference.
**Future Work**

**On matching**

Term-based matching using normal inferences is

- **nice** — it is firmly based on the theory presented here; and
- **problematic** — the pilot-version, we are currently testing, is horribly 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.
- **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 — Illustrated
- The SWITCH rule
- Proof Method
**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.)*
**Definition (2 kinds of placing)**

Merge \( merge_n : \langle n, \emptyset, \emptyset \rangle \rightarrow \langle 1, [\emptyset], \emptyset \rangle \)

Permutation \( \pi \quad \pi_{\vec{X}} : \langle m, \vec{X}, X \rangle \rightarrow \langle m, \pi(\vec{X}), X \rangle \)

**(Examples) Merge and permutation**

\[
merge_3 = \begin{array}{ccc}
0 & 1 & 2 \\
\end{array}
\]

\[
\{0 \mapsto 2, 1 \mapsto 0, 2 \mapsto 1\}_{\{x\}, \emptyset, \{y\}} = \begin{array}{ccc}
1 & 2 & 0 \\
y & y & x \\
\end{array}
\]
**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 \( /X : \langle 0, [], X \rangle \rightarrow \text{id}_\varepsilon \)

**EXAMPLES) LINKINGS**

Plain substitution \([y_1, y_2, y_3]/[\{x_1, x_2\}, \{\}, \{x_3\}] = \)

Renaming \( \alpha \) \([y_1, y_2, y_3]/[x_1, x_2, x_3] = \)

Closure \( /\{x_1, x_2, x_3\} = \)

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

Concretion  
\( \llbracket X \rrbracket : \langle 1, [X], X \rangle \rightarrow \langle 1, [\emptyset], X \rangle \)

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

(Examples) Concretions and Ions

\( \llbracket \{x_1, x_2\} \rrbracket = \)

\( K_{[y_1, y_2]}([\{x_1\}, \{x_2, x_3\}, \{\}]) = \)
(Standard categorical composition and tensor product as defined earlier.)

**Definition (Abstraction)**

\[
\text{Abstraction } (Y)P : I \rightarrow \langle 1, [Y], Z \uplus Y \rangle
\]

**Example Abstraction**

\[
(\{y_1, y_2\})(\{y_3\})^{-1} \{y_1, y_2, y_3, z\}^{-1} = \begin{array}{c}
y_1 \ y_2 \ y_3 \ y_3 \\
\hline
0 \\
y_1 \ y_2 \ y_3 \ y_3 \\
\end{array}
\]
BIGRAPH $A$ — A MODEL OF AN OFFICE

\[ A = \]

```
secret
server

pc  pda  pda
office
```
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))**

\[
\begin{align*}
\text{MDNF} & \quad M := (K_{\vec{y}}(\vec{x}) \otimes \text{id}_Z)P \\
\text{PDNF} & \quad P := (Y^B) (\text{merge}_{n+k} \otimes \text{id}_Y) \left( (\bigotimes_i^n \alpha_i \bigotimes M_i) \right) \pi \\
\text{DDNF} & \quad D := (P_0 \otimes \ldots \otimes P_{n-1}) \pi \otimes \alpha \\
\text{BDNF} & \quad B := (\omega \otimes \bigotimes_i^n \vec{\sigma}_i) D
\end{align*}
\]

More details for each form on the next pages.
Free Discrete Molecules — MDNF

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

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

where $P : I \rightarrow \langle 1, (\{\vec{X}\}), \{\vec{X}\} \cup Z \rangle$ 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].)
Any name-discrete prime $P : I \rightarrow \langle 1, Y^B, Y \rangle$ can be expressed as

$$ (Y^B) \left( merge_{n+k} \otimes id_Y \right) \left( \Gamma \alpha_0 \otimes \cdots \otimes \Gamma \alpha_{n-1} \otimes M_0 \otimes \cdots \otimes M_{k-1} \right) \pi $$

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

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 the sites inside the primes; the permutation $\pi$ changes accordingly to preserve the innerface.

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

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

where $D : I \rightarrow \langle n, \tilde{X}, \{ \tilde{X} \} \cup Z \rangle$ is name-discrete, $\omega : Z \rightarrow Y^F$ is a (global) wiring, and each $\hat{\sigma}_i : (\tilde{X}_i) \rightarrow (Y^B_i)$ 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].)
This section contains illustrated versions of the rules.
$$
\begin{array}{c}
\text{CLOSE} \quad \frac{\sigma_a, \sigma_R, \text{id}_{Y_R} \otimes \sigma_C \vdash a, R \hookrightarrow C, d}{\sigma_C :\rightarrow Y \cup Y_C \quad \sigma_R :\rightarrow U \cup Y_R} \\
(id \otimes / (Y_R \cup Y_C))\sigma_a, (id \otimes / Y_R)\sigma_R, (id \otimes / Y_C)\sigma_C \vdash a, R \hookrightarrow C, d
\end{array}
$$
**Conclusion**

\[
\begin{align*}
\text{CLOSE} & : \qquad \sigma_a, \sigma_R, \text{id}_Y \otimes \sigma_C \vdash a, R \leftrightarrow C, d \quad \sigma_C : \rightarrow Y \cup Y_C \quad \sigma_R : \rightarrow U \cup Y_R \\
& \quad \left(\text{id} \otimes / (Y_R \cup Y_C)\right)\sigma_a, \left(\text{id} \otimes / Y_R\right)\sigma_R, \left(\text{id} \otimes / Y_C\right)\sigma_C \vdash a, R \leftrightarrow C, d
\end{align*}
\]

Then

\[
\begin{align*}
\text{id} & \\
\text{id} & \\
\text{\(s_a\)} & \\
\text{id} & \\
\text{\(\eta_C\)} & \\
\text{id} & \\
\text{\(s_C\)} & \\
\text{id} & \\
\text{\(\eta_R\)} & \\
\text{id} & \\
\text{\(s_R\)} & \\
\text{id} & \\
\text{id} & \\
\text{id} & \\
\text{id} & \\
\text{id} & \\
\text{\(d\)} & \\
\text{\(a\)} & \\
\text{\(C\)} & \\
\text{\(R\)} & \\
\end{align*}
\]
\[
\omega_a, \omega_R, \omega_C \vdash a, \bigotimes_i^m P_{\pi^{-1}(i)} \leftrightarrow C, (\pi \otimes \text{id})d \\
\omega_a, \omega_R, \omega_C \vdash a, \bigotimes_i^m P_i \leftrightarrow C_\pi, d \\
\]

If $\rho = \pi^{-1}$

\[
\text{id} \quad \text{id}_Z \\
\text{id} \quad \text{id}_Z \\
\text{w}_a \quad \text{w}_c \\
C \quad \text{w}_R \\
P_{\text{\texttt{\textbackslash rho}}(1)} \quad \ldots \quad P_{\text{\texttt{\textbackslash rho}}(n)} \\
\overline{\pi} \times \text{id} \\
d 
\]
**RULES — DETAILS (CONT.)**

**CONCLUSION**

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

\[
\omega_a, \omega_R, \omega_C \vdash \bigotimes_i^m P_i \leftrightarrow C\pi, d
\]

Then

\[
\begin{aligned}
\text{id} & \quad \cdots & \quad \text{id} \\
\text{F} \quad \cdots & \quad \text{F} \\
\text{id}_\pi & \quad \cdots & \quad \text{id}_\pi \\
\text{P}_1 & \quad \cdots & \quad \text{P}_n \\
\text{d} & \quad & \\
\end{aligned}
\]
RULES — DETAILS (cont.)

Premise

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

If

\[
\begin{align*}
& \quad \text{id} \quad \text{w}_a = \quad \text{id} \quad \text{w} \quad \text{w}_C \\
& \quad \text{id} \quad \text{w}_R \quad \text{id}_Zd = \quad \text{id} \quad \text{w}_b \\
& \quad \text{id} \quad \text{w}_D = \quad \text{id} \quad \text{w}_S \quad \text{id}_Ze
\end{align*}
\]

\[
\begin{align*}
& \quad \text{a} \\
& \quad \text{R} \quad \text{d} \\
& \quad \text{b} \\
& \quad \text{S} \quad \text{e}
\end{align*}
\]
RULES — DETAILS (CONT.)

CONCLUSION

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

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

Then

\[
\begin{align*}
\begin{array}{ccc}
\text{id} & \wedge_a & \wedge_b \\
\end{array} =
\begin{array}{ccc}
\text{id} & \wedge_C & \wedge_D \\
\end{array}
\]

\[
\begin{array}{ccc}
\text{id} & \wedge_Zd & \wedge_R & \wedge_S & \wedge_Ze \\
\text{a} & \text{id} & \text{b} & \text{R} & \text{S} & \text{d} & \text{e} \\
\end{array}
\]

\[
\begin{array}{ccc}
\text{id} & \wedge_C & \wedge_D \\
\end{array}
\]

\[
\begin{array}{ccc}
\text{id} & \wedge_Zd & \wedge_R & \wedge_S & \wedge_Ze \\
\text{a} & \text{id} & \text{b} & \text{R} & \text{S} & \text{d} & \text{e} \\
\end{array}
\]

\[
\begin{array}{ccc}
\text{id} & \wedge_C & \wedge_D \\
\end{array}
\]

TROELS C. DAMGAARD (ITU)  SYNTACTIC THEORY FOR BIGRAPHS  ENS, PARIS  58 / 61
RULES — DETAILS (CONT.)

Premise

\[ \sigma_a \otimes \omega_a, \omega_R, \sigma_C \otimes \omega_C \vdash p, R \leftrightarrow P, d \quad \sigma_a : Z \rightarrow W \quad \sigma_C : U \rightarrow W \]

\[ \omega_a, \omega_R, \omega_C \vdash (\widehat{\sigma_a} \otimes \text{id})(Z)p, R \leftrightarrow (\widehat{\sigma_C} \otimes \text{id})(U)P, d \]
RULES — DETAILS (cont.)

CONCLUSION

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

Then

\[
\begin{align*}
\text{id} & \quad \omega_a \quad \text{id} \\
(s_a) & \quad \text{id} \quad (s_C) \\
(Z) & \quad p \quad (U) \quad P \\
R & \\
d & \quad \text{id}_Z
\end{align*}
\]
RULES — DETAILS (CONT.)

Premise

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

If

\[
\begin{array}{c}
\text{id} \\
\text{w}_a
\end{array} = \begin{array}{c}
\text{id} \\
\text{w}_c
\end{array} = \begin{array}{c}
\text{C} \\
\text{w}_R
\end{array} = \begin{array}{c}
\text{R} \\
\text{id}_Z
\end{array} = \begin{array}{c}
\text{a} \\
\text{d}
\end{array}
\]
RULES — DETAILS (cont.)

**Conclusion**

\[
\begin{align*}
\text{MERGE} & \\
\omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d & \quad \text{aglobal} \\
\omega_a, \omega_R, \omega_C \vdash (merge \otimes id)a, R \leftrightarrow (merge \otimes id)C, d
\end{align*}
\]

Then

\[
\begin{align*}
\text{id} & \quad w_a \\
\text{merge} & \quad \text{id} \\
\text{merge} & \quad \text{id} \\
\text{merge} & \quad \text{id} \\
\text{id} & \quad w_R \\
\text{id} & \quad \text{id}_Z
\end{align*}
\]

\[
\begin{align*}
a & \\
R & \\
d &
\end{align*}
\]
**Rules — Details (cont.)**

**Premise**

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

If

\[
\begin{align*}
\text{id} & \quad \quad \omega_a \\
(s_X) & \quad \quad \text{id} \\
\hline
\hline
p & \quad \quad \omega_C
\end{align*}
= \quad
\begin{align*}
\text{id} & \quad \quad \omega_R \\
(s_Z) & \quad \quad \text{id} \\
\hline
\hline
P & \quad \quad \text{id} \\
R & \quad \quad \text{id}_Z
\end{align*}
\]

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

\[
\begin{array}{c}
\text{id} \\
\text{s} \\
\text{w_a} \\
\text{K_y(X)} \\
\text{id} \\
\text{p}
\end{array}
\quad =
\quad
\begin{array}{c}
\text{id} \\
\text{s} \\
\text{\alpha} \\
\text{w_C} \\
\text{id} \\
\text{w_R} \\
\text{P} \\
\text{R} \\
\text{id_Z} \\
\text{d}
\end{array}
\]
\[
\begin{align*}
\text{PREMISE} & \quad \omega_a, \text{id}_e, \omega_C(\sigma \otimes \omega_R \otimes \text{id}) \vdash p, \text{id} \leftrightarrow P, d \\
& \quad P :\to \langle W \cup Y \rangle \\
& \quad \sigma : W \to U
\end{align*}
\]

\[
\begin{align*}
\omega_a, \omega_R, \omega_C \vdash p, (\hat{\sigma} \otimes \text{id})(W)P & \leftrightarrow \Gamma U \setminus d
\end{align*}
\]
RULES — DETAILS (CONT.)

**CONCLUSION**

\[
\begin{align*}
\text{SWITCH} & \quad \omega_a, \text{id}_\epsilon, \omega_C (\sigma \otimes \omega_R \otimes \text{id}) \vdash p, \text{id} \leftrightarrow P, d \quad P : \rightarrow \langle W \sqcup Y \rangle \quad \sigma : W \rightarrow U \\
\omega_a, \omega_R, \omega_C & \vdash p, (\hat{\sigma} \otimes \text{id})(W)P \leftrightarrow \lceil U \rceil, d
\end{align*}
\]
The switch rule

All rules work, intuitively, by comparing structure in the agent and context position

\[ \omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d. \]
The Switch Rule

All rules work, intuitively, by comparing structure in the agent and context position

$$\omega_a, \omega_R, \omega_C \vdash a, R \leftrightarrow C, d.$$

A single rule, `SWITCH`, allows us (essentially) to switch redex structure to the context position, when the context contains only a hole.

$$\text{SWITCH}$$

\[
\begin{align*}
\omega_a, \text{id}_\epsilon, \omega_C(\sigma \otimes \omega_R \otimes \text{id}) & \vdash p, \text{id} \leftrightarrow P, d \\
P & \vdash \langle W \uplus Y \rangle \\
\sigma & : W \rightarrow U \\
\omega_a, \omega_R, \omega_C & \vdash p, (\hat{\sigma} \otimes \text{id})(W)P \leftrightarrow \lceil \alpha \rceil, d
\end{align*}
\]
THE SWITCH RULE

All rules work, intuitively, by comparing structure in the agent and context position

\[ \omega_a, \omega_R, \omega_C \vdash a, \text{id} \leftrightarrow R, d. \]

A single rule, \text{SWITCH}, allows us (essentially) to \textit{switch} redex structure to the context position, when the context contains only a hole.

\[
\text{SWITCH} \quad \omega_a, \text{id}_\epsilon, \omega_C (\sigma \otimes \omega_R \otimes \text{id}) \vdash p, \text{id} \leftrightarrow P, d \quad P : \rightarrow \langle W \uplus Y \rangle \quad \sigma : W \rightarrow U
\]

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

Intuitively, \text{SWITCH} allows us to

- build inferences by initially seeking to match agent and context; and,
- when reaching a \textit{hole} in the context, \textit{switch} redex structure to the context position in the matching sentence.
**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 CLOSE and the PERM rule on a valid sentence, of equal size, of the form $\sigma_a, \sigma_R, \sigma_C \models a, S \leftrightarrow \bigotimes_i P_i, e$.

**Lemma (2)**

Every valid sentence $\sigma_a, \sigma_R, \sigma_C \models a, R \leftrightarrow P \otimes \bigotimes_i P_i, d$, with $P$ and $P_i$ prime and discrete, is provable using the PRODUCT rule on valid sentences, of lesser or equal size, of the form $\sigma_a^P, \sigma_R^P, \sigma_C^P \parallel \sigma_S \models p, S \leftrightarrow P, e$ and $\sigma_a^C, \sigma_R^C, \sigma_C^C \parallel \sigma_S \models a', R' \leftrightarrow \bigotimes_i P_i, e'$.
PROOF METHOD (COMPLETENESS) (CONT.)

**Lemma (3)**

Every valid sentence \( \sigma_a, \sigma_R, \sigma_C \models a, R \hookrightarrow \text{id}_\epsilon, d \) is provable using **PRODUCT** and **WIRING-AXIOM**.

**Lemma (4)**

Every valid sentence \( \omega_a, \omega_R, \omega_C \models p, R \hookrightarrow P, d \), with \( p \) and \( P \) prime and discrete, is provable using the **LSUB** rule on a valid sentence, of lesser or equal size, of the form \( \omega'_a, \omega'_R, \omega'_C \models p', R \hookrightarrow P', d \), where \( p' \) and \( P' \) are discrete free primes.
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 PRIME-AXIOM 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 $n$. □