A Graph Rewriting Calculus: Applications to Biology and Autonomous Systems

Oana Andrei

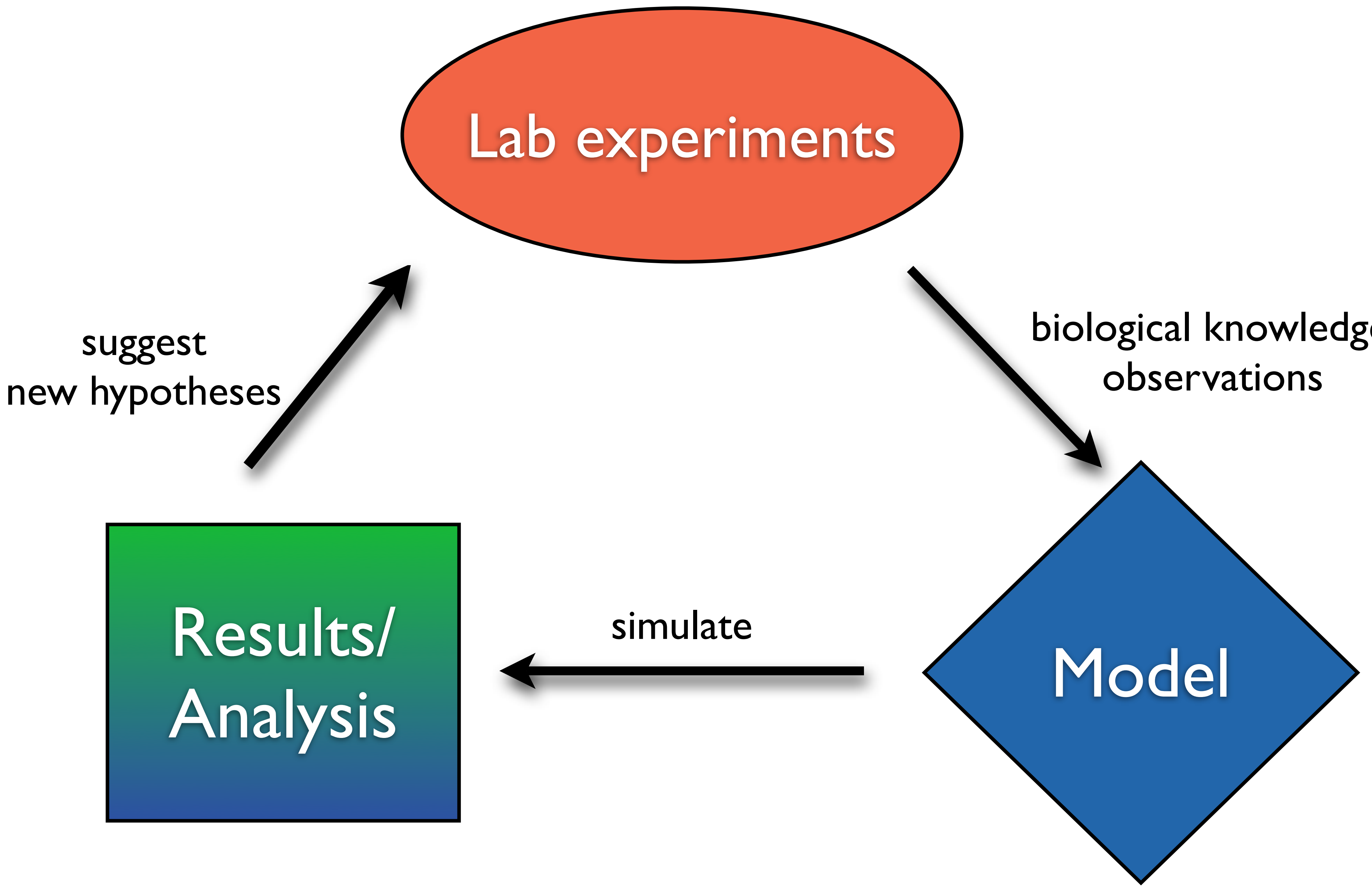
PhD Defense

Institut National Polytechnique de Lorraine
INRIA Nancy - Grand-Est & LORIA

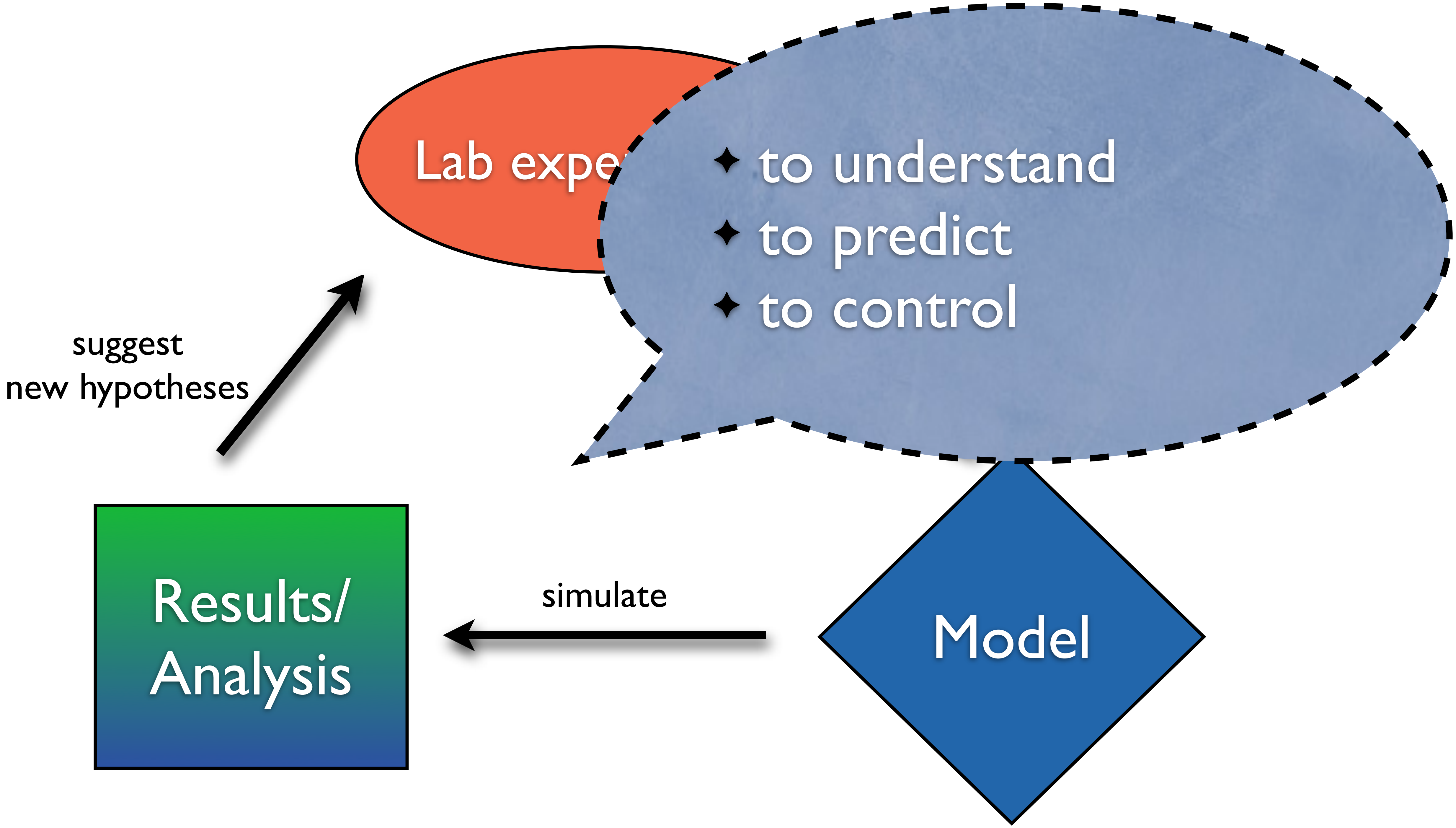
Adviser: Hélène Kirchner

Thanks to Pareo Team, especially to Horatiu Cirstea

Formal Methods in Systems Biology



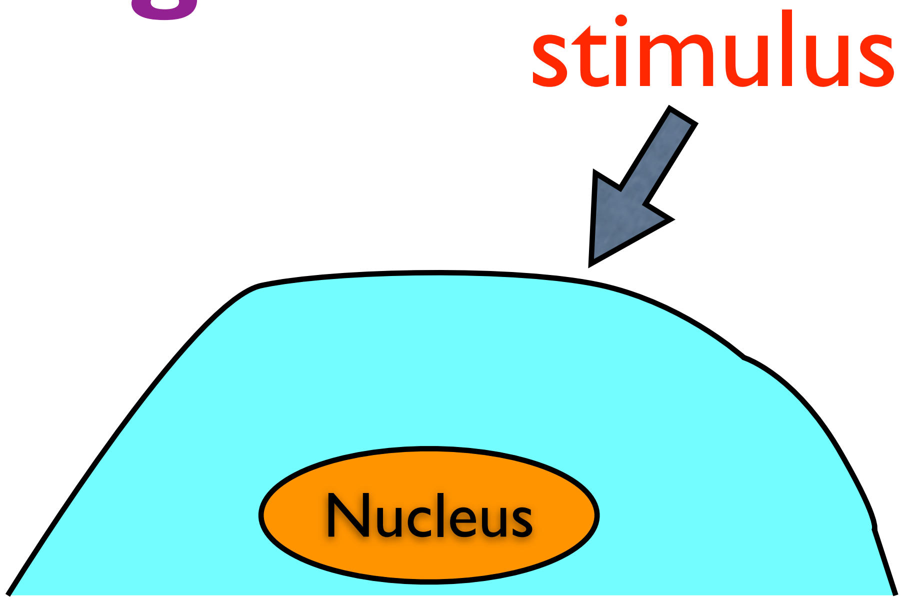
Formal Methods in Systems Biology



Cell Signalling

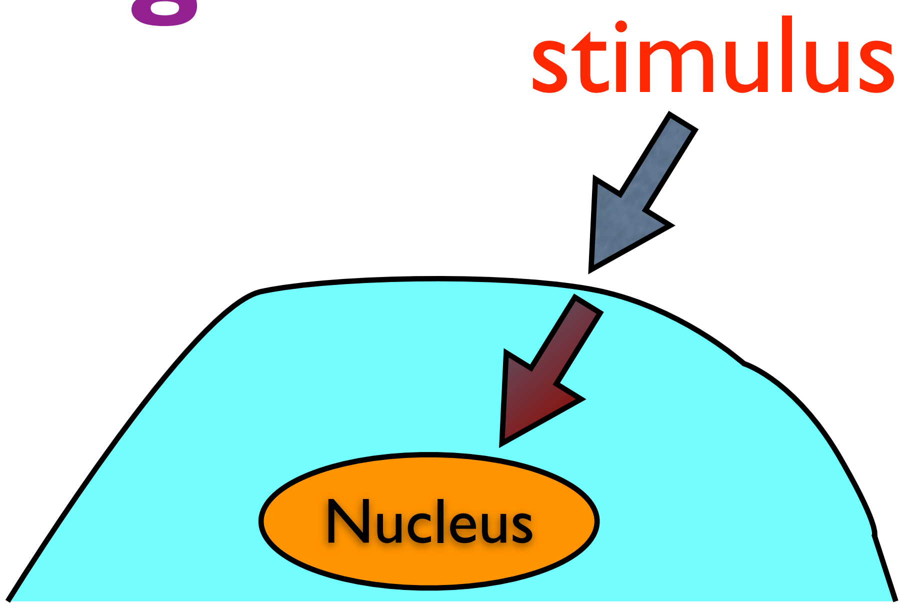
- communication between cells
- *cellular processes:* proliferation, cell growth, programmed cell death...
- *malfunctions:* cancer, diabetes, autoimmunity...

Cell Signalling



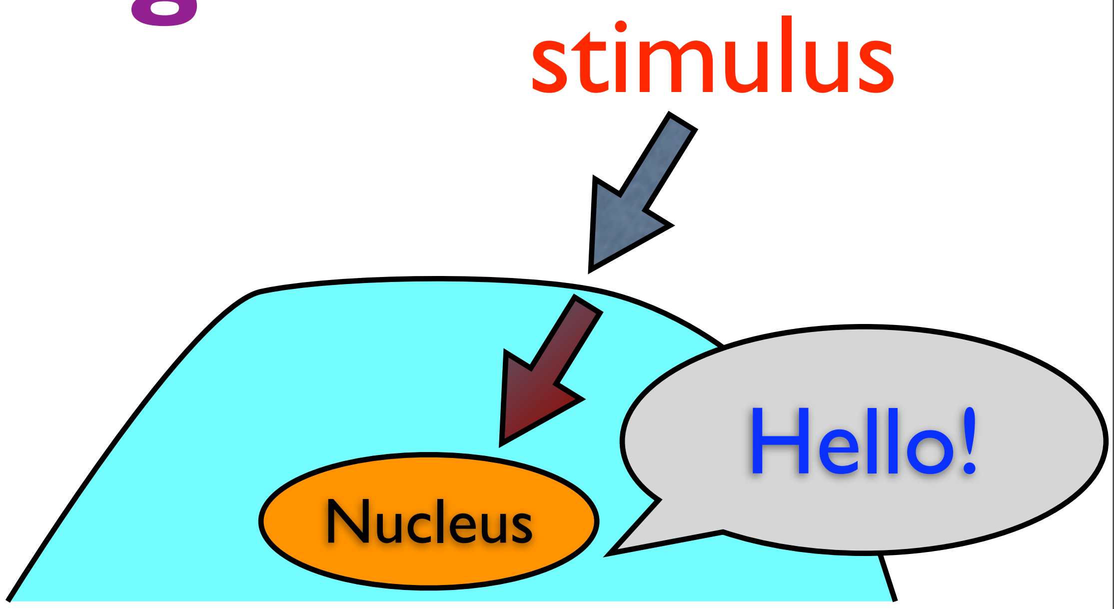
- communication between cells
- *cellular processes:* proliferation, cell growth, programmed cell death...
- *malfunctions:* cancer, diabetes, autoimmunity...

Cell Signalling



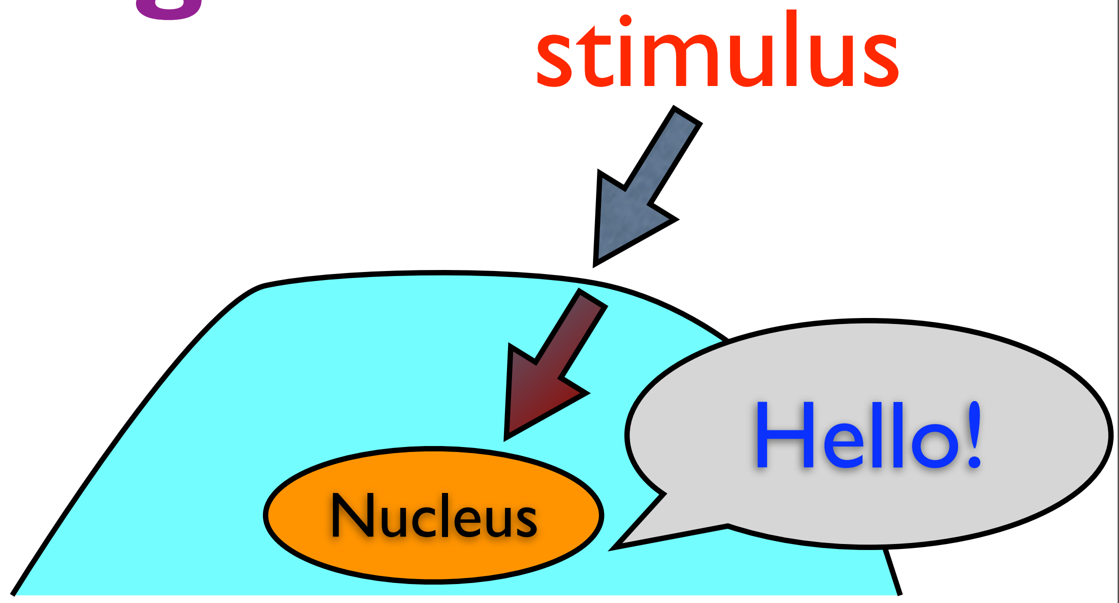
- communication between cells
- *cellular processes:* proliferation, cell growth, programmed cell death...
- *malfunctions:* cancer, diabetes, autoimmunity...

Cell Signalling



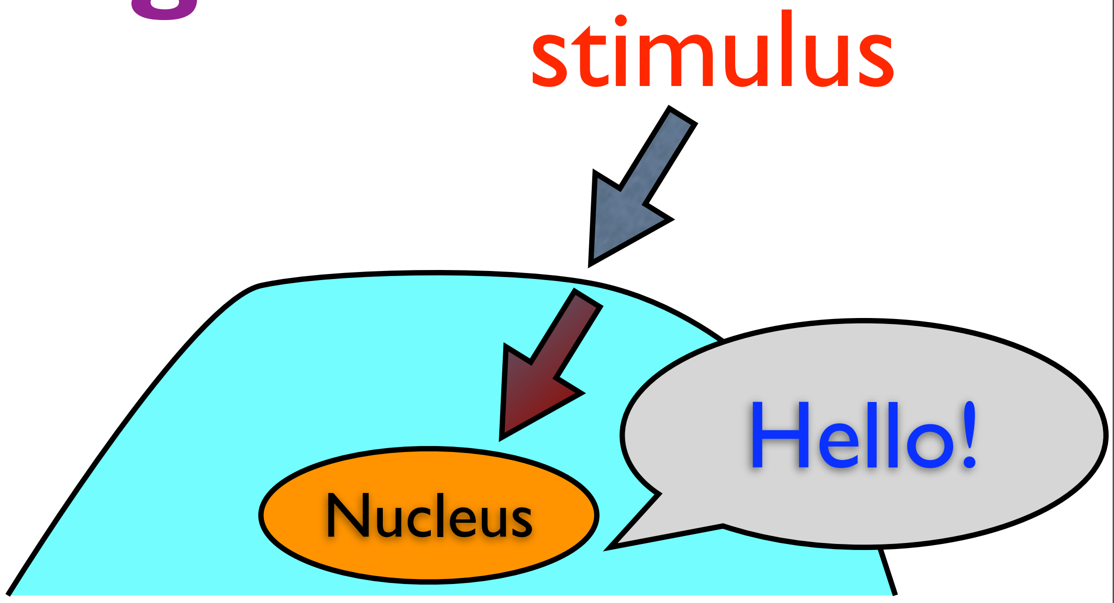
- communication between cells
- *cellular processes:* proliferation, cell growth, programmed cell death...
- *malfunctions:* cancer, diabetes, autoimmunity...

Cell Signalling



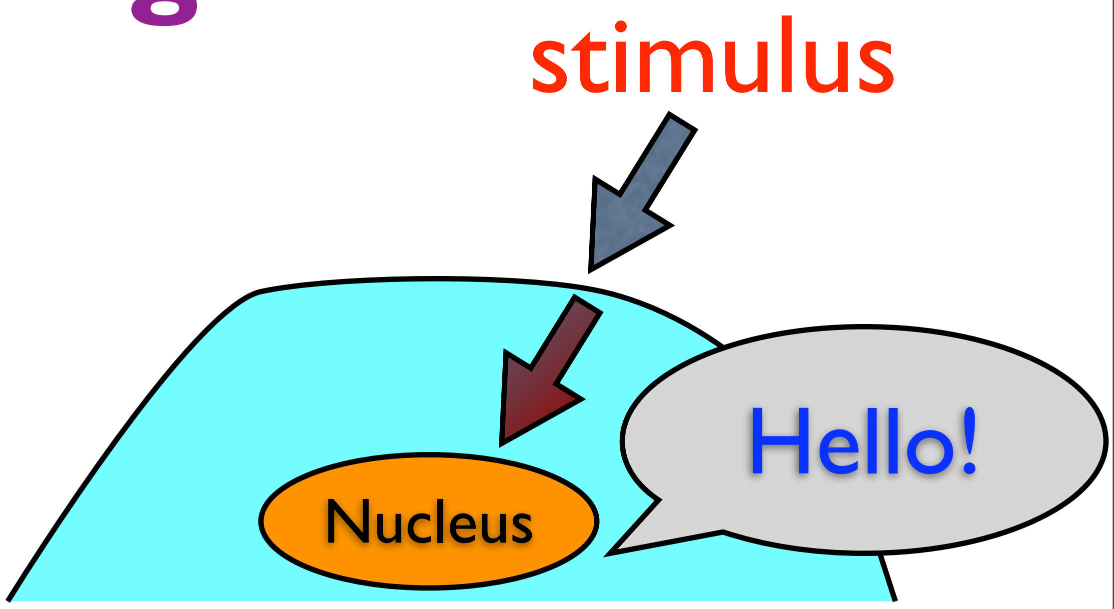
- communication between cells
- *cellular processes:* proliferation, cell growth, programmed cell death...

Cell Signalling



- communication between cells
- *cellular processes:* proliferation, cell growth, programmed cell death...
- *malfunctions:* cancer, diabetes, autoimmunity...

Cell Signalling



- communication between cells
- *cellular processes:* proliferation, cell growth, programmed cell death...
- *malfunctions:* cancer, diabetes, autoimmunity...

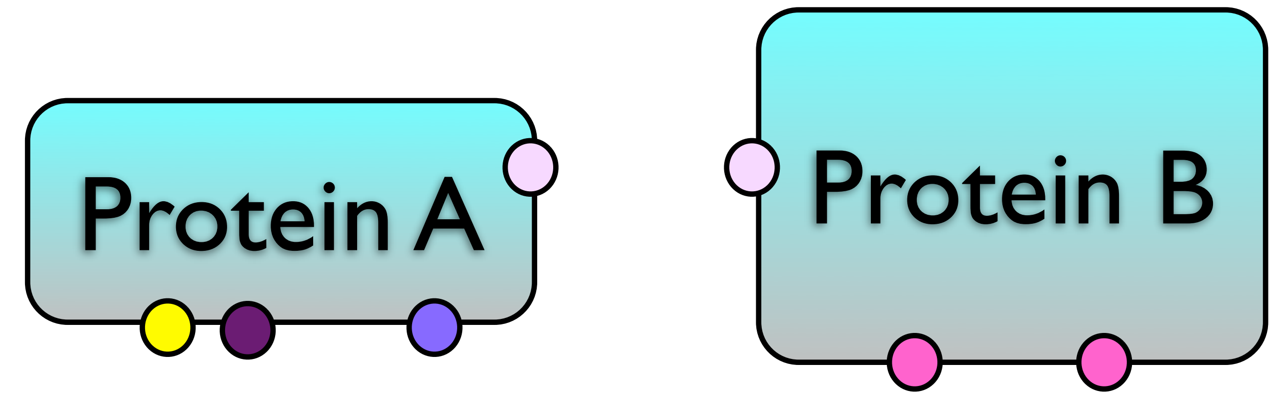
Need of good, predictive models for guiding experimentations and drug development.

Autonomous Systems

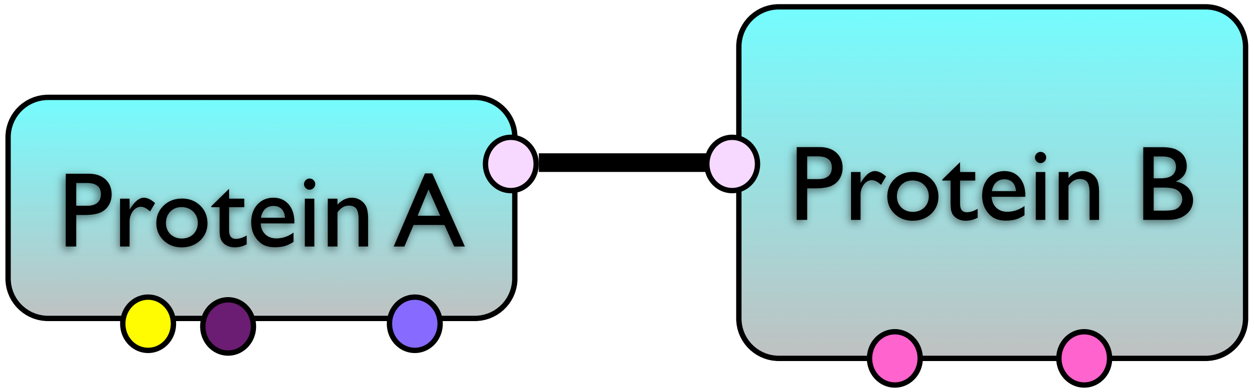
- *Living cell are extremely well-organized autonomous systems. [Cardelli 05]*
- Autonomic computing [KephartChess 03] refers to self-manageable systems initially provided with some high-level instructions from administrators.

➡ Bio-inspired modeling of distributed systems

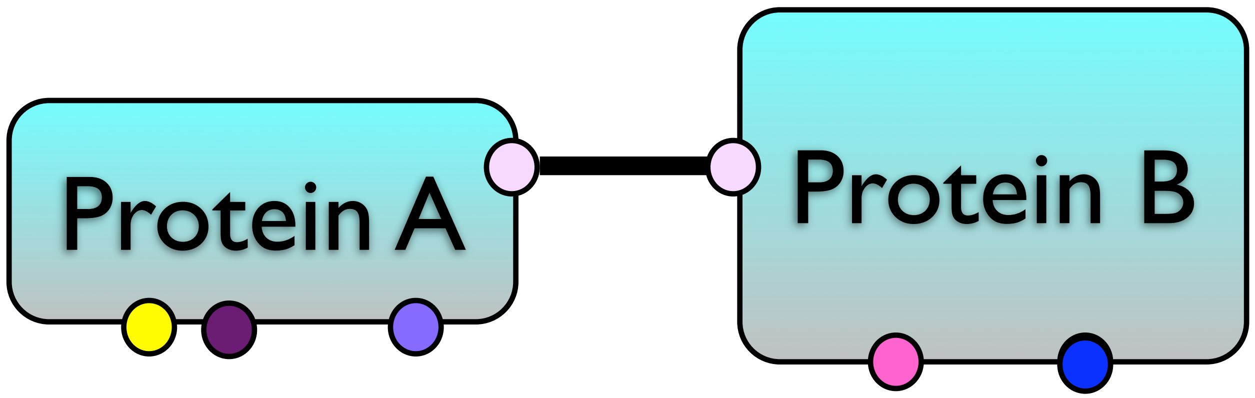
Molecular Complexes as Graphs



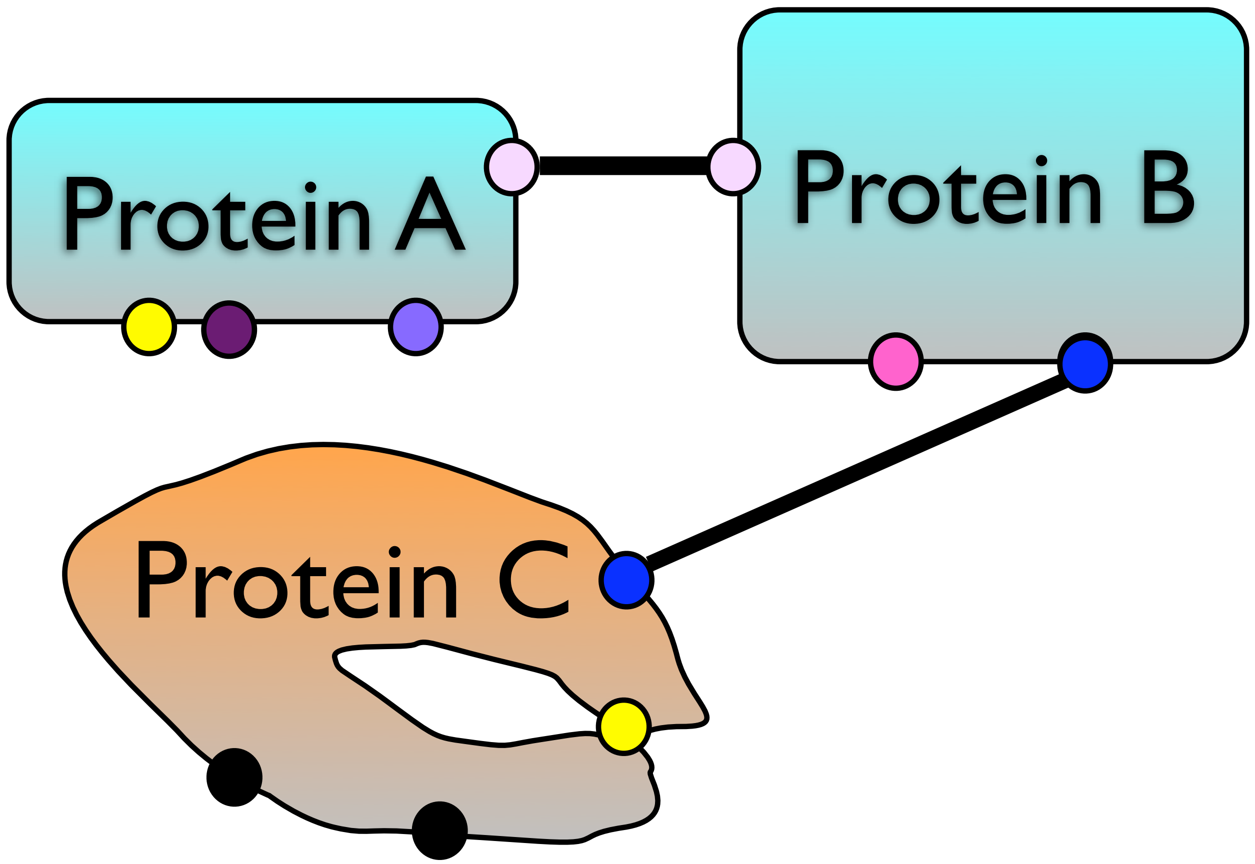
Molecular Complexes as Graphs



Molecular Complexes as Graphs



Molecular Complexes as Graphs



Rule-based Modeling

- well-suited for modeling bio-molecular interactions, cell-signaling
- a rule $l \rightarrow r$ defines a class of reactions
- rewrite strategies control the rule application

Background Works

- the graph structure of molecular complexes in the κ-calculus [DanosLaneve 04], BioNetGen [Faeder et al. 05]
- rule-based modeling of a chemical reactor [Bournez et al. 03]

Background Works

- the chemical model of computation, Υ-calculus, a higher-order chemical calculus [Banatre et al. 05]:
  - ✦ a chemical solution where molecules interact freely according to reaction rules,
  - ✦ everything is a molecule

$$prod = \textbf{replace}\ X,\ Y\ \textbf{by}\ X \times Y$$

$$\langle prod, 3, 1, 4, 5, 2\rangle \ \rightarrow\ \langle prod, 1, 4, 15, 2\rangle \ \rightarrow^{*}$$

$$\langle prod,\ 120\rangle$$

Background Works

- the rewriting calculus [CirsteaKirchner 01] :
  - ✦ extends first-order term rewriting and the λ-calculus
  - ✦ all the basic ingredients of rewriting are explicit objects of the calculus

$$(s(x)+y \rightarrow s(x+y))\ (s(5)+s(2)) \longrightarrow_{\rho} s(5+s(2))$$

Objective of This Thesis

- to develop a calculus based on graph rewriting for describing molecules, reactions, and biochemical network generation
- towards a **biochemical calculus**:
  - ★ add a biological flavor to the chemical calculus
  - ★ rewrite rules and rewrite strategies
  - ★ property verification for the modeled systems

Outline

- An Abstract Biochemical Calculus
- Port Graph Rewriting
- A Biochemical Calculus Based on Strategic Port Graph Rewriting
- Runtime Verification in the Biochemical Calculus
- Conclusions and Perspectives

- An Abstract Biochemical Calculus
- Port Graph Rewriting
- A Biochemical Calculus Based on Strategic Port Graph Rewriting
- Runtime Verification in the Biochemical Calculus
- Conclusions and Perspectives

An Abstract Biochemical Calculus

-- the $\rho_{\langle\Sigma\rangle}$-calculus --

- a higher-order rewriting calculus
- first citizens:
  - structured objects
  - rules (abstractions)
  - rule applications
- strategies as objects
- encompasses the rewriting calculus
- generalizes the ϒ-calculus (abstractions over a class of patterns, not only variables)

[AndreiKirchner08a,b,c]

Basic Syntax

- structured objects $\mathcal{O}$ over a monoidal category Σ with _ _ an associative and commutative product tensor and ε neutral element (eq. theory $\mathbb{T}$)
- iterative construction of abstractions and abstract molecules

Basic Syntax

Ex.: the category of graphs **Graph**

- structured objects $\mathcal{O}$ over a monoidal category Σ with _ _ an associative and commutative product tensor and ε neutral element (eq. theory $\mathbb{T}$)
- iterative construction of abstractions and abstract molecules

Basic Syntax

Ex.: the category of graphs **Graph**

- structured objects $\mathcal{O}$ over a monoidal category Σ with _ _ an associative and commutative product tensor and ε neutral element (eq. theory $\mathbb{T}$)
- iterative construction of abstractions and abstract molecules

Basic Syntax

Ex.: the category of graphs **Graph**

- structured objects $\mathcal{O}$ over a monoidal category Σ with _ _ an associative and commutative product tensor and ε neutral element (eq. theory $\mathbb{T}$)

- iterative construction of abstractions and abstract molecules

$$
\begin{array}{rcl}
\mathcal{A}_0 & ::= & \mathcal{O} \Rightarrow \mathcal{O} \\
& | & (\mathcal{O} \Rightarrow \mathcal{O}) \Rightarrow (\mathcal{O} \Rightarrow \mathcal{O}) \\
\mathcal{M}_0 & ::= & \mathcal{O} \mid \mathcal{A}_0 \mid \mathcal{M}_0\ \mathcal{M}_0
\end{array}
$$

Basic Syntax

Ex.: the category of graphs **Graph**

- structured objects $\mathcal{O}$ over a monoidal category Σ with … and commutative product te… ent (eq. theory $\mathbb{T}$)

Ex.: $G_1 \Rightarrow G_2$ is a graph morphism, arrow in **Graph**

- it… bstractions and abstract m…

$$
\begin{array}{rcl}
\mathcal{A}_0 & ::= & \mathcal{O} \Rightarrow \mathcal{O} \\
& | & (\mathcal{O} \Rightarrow \mathcal{O}) \Rightarrow (\mathcal{O} \Rightarrow \mathcal{O}) \\
\mathcal{M}_0 & ::= & \mathcal{O} \mid \mathcal{A}_0 \mid \mathcal{M}_0\ \mathcal{M}_0
\end{array}
$$

Basic Syntax

Ex.: the category of graphs **Graph**

- structured objects $\mathcal{O}$ over a monoidal category Σ with _ _ an associative and commutative product tensor and ε neutral element (eq. theory $\mathbb{T}$)
- iterative construction of abstractions and abstract molecules

$$
\begin{array}{rcl}
\mathcal{A}_0 & ::= & \mathcal{O} \Rightarrow \mathcal{O} \\
& | & (\mathcal{O} \Rightarrow \mathcal{O}) \Rightarrow (\mathcal{O} \Rightarrow \mathcal{O}) \\
\mathcal{M}_0 & ::= & \mathcal{O} \mid \mathcal{A}_0 \mid \mathcal{M}_0\ \mathcal{M}_0
\end{array}
$$

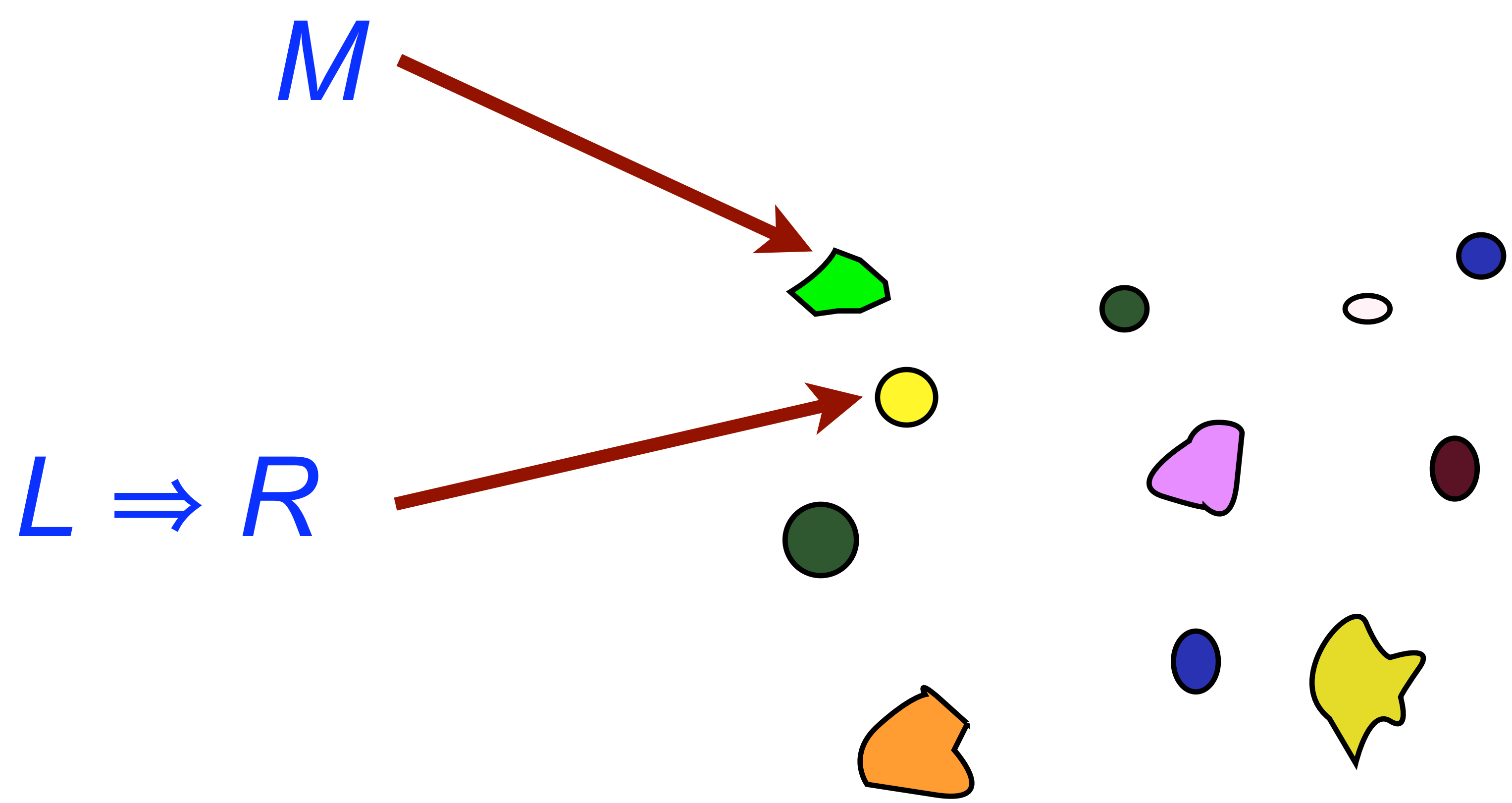
Basic Syntax

Ex.: the category of graphs **Graph**

- structured objects $\mathcal{O}$ over a monoidal category Σ with _ _ an associative and commutative product tensor and ε neutral element (eq. theory $\mathbb{T}$)
- iterative construction of abstractions and abstract molecules

$$
\begin{array}{lcl}
\mathcal{A}_0 & ::= & \mathcal{O} \Rightarrow \mathcal{O} \\
 & \mid & (\mathcal{O} \Rightarrow \mathcal{O}) \Rightarrow (\mathcal{O} \Rightarrow \mathcal{O}) \\
\mathcal{M}_0 & ::= & \mathcal{O} \mid \mathcal{A}_0 \mid \mathcal{M}_0\ \mathcal{M}_0
\end{array}
\qquad
\begin{array}{lcl}
\mathcal{A} & ::= & \mathcal{A}_0 \mid \mathcal{O} \Rightarrow \mathcal{M}_0 \\
\mathcal{M} & ::= & \mathcal{X} \mid \mathcal{M}_0 \mid \mathcal{A} \mid \mathcal{M}\ \mathcal{M}
\end{array}
$$

Interactions

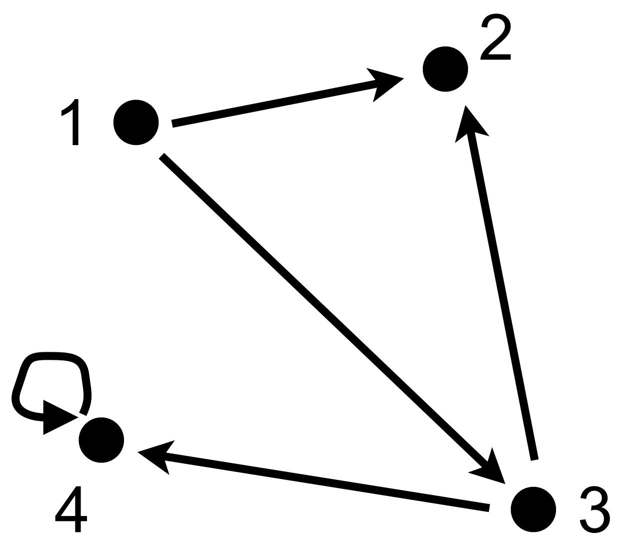


Is $L \Rightarrow R$ applicable to M ?

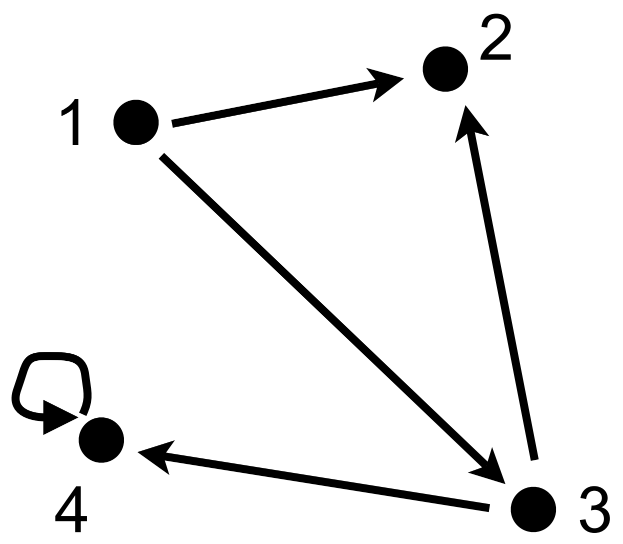
Matching

- submatching equation modulo $\mathbb{T}$: $L \lll_{\mathbb{T}} M$
- solutions have the form $(\sigma, M^-, \mathcal{B})$ such that $M =_{\mathbb{T}} M^-\lfloor\sigma(L)\rfloor_{\mathcal{B}}$
- a submatching algorithm for every instantiation of Σ and $\mathbb{T}$

Matching and Decomposition

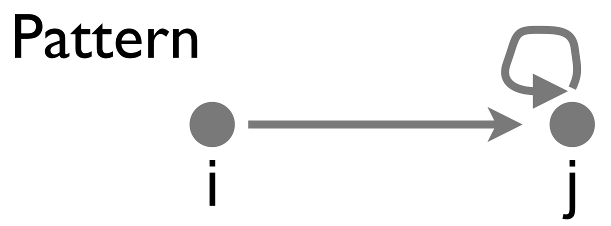


Matching and Decomposition

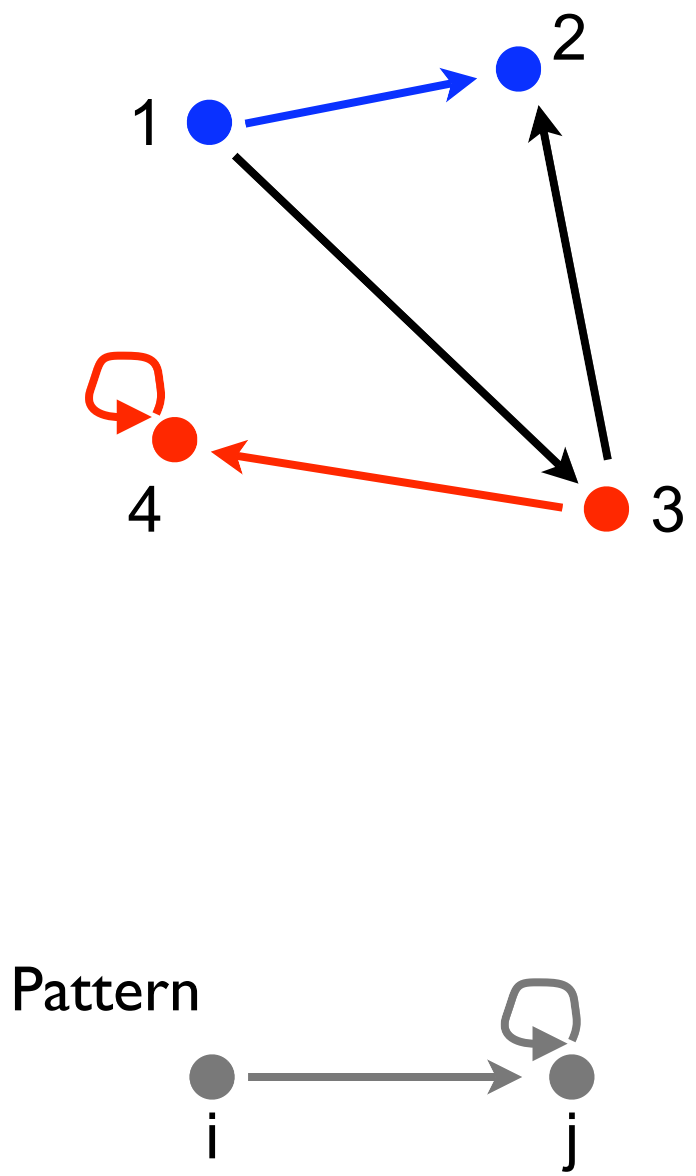


Pattern

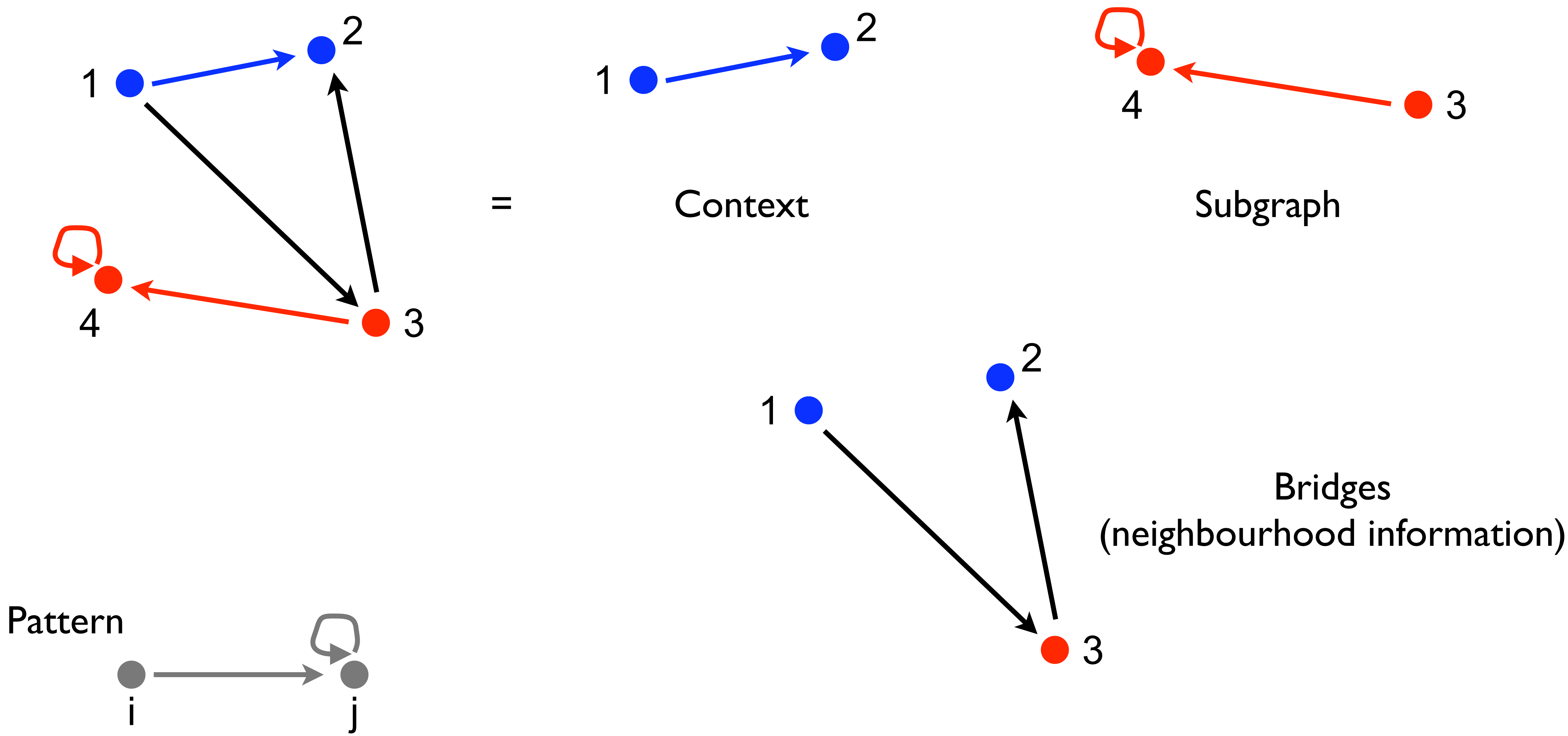
Matching and Decomposition



Matching and Decomposition



Matching and Decomposition



Worlds and Multiverses

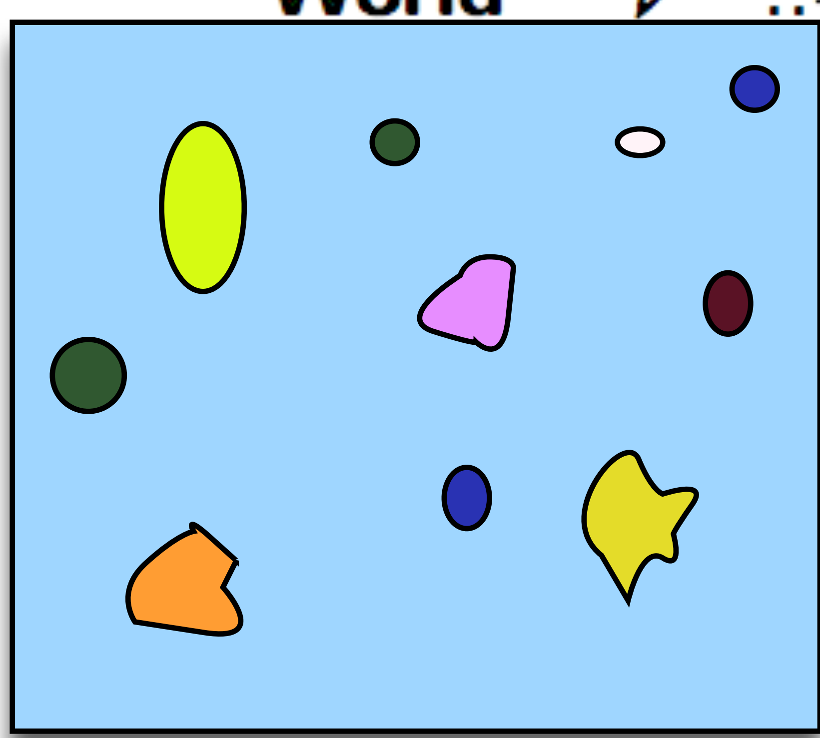
- the molecules in an environment are encapsulated in a world
- alternative worlds are grouped in a multiverse

$$
\begin{array}{llll}
\textbf{World} & \mathcal{V} & ::= & \mathcal{Y} \mid [\mathcal{M}] \\
\textbf{Multiverse} & \mathcal{W} & ::= & \mathcal{Z} \mid \{\mathcal{V} \ldots \mathcal{V}\} \mid \{\mathcal{W} \ldots \mathcal{W}\}
\end{array}
$$

Worlds and Multiverses

- the molecules in an environment are encapsulated in a world
- alternative worlds are grouped in a multiverse

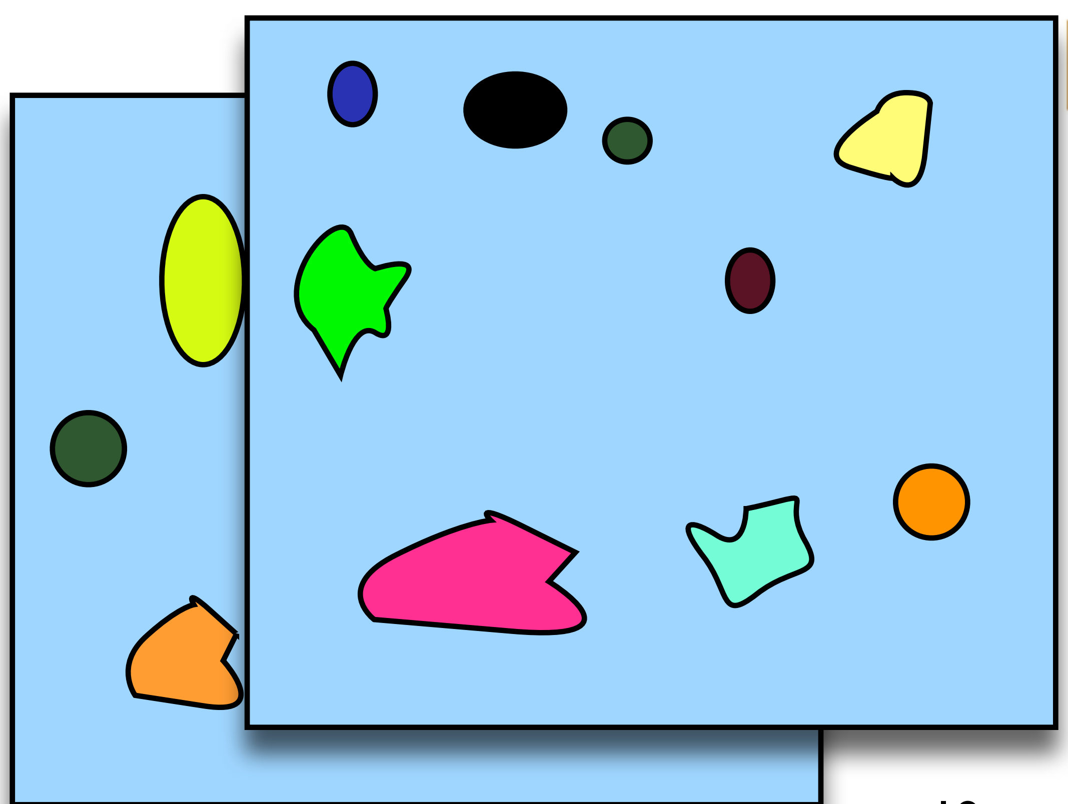
World $\mathcal{V} \; ::= \; \mathcal{Y} \mid [\mathcal{M}]$



$= \; \mathcal{Z} \mid \{\mathcal{V}\ldots\mathcal{V}\} \mid \{\mathcal{W}\ldots\mathcal{W}\}$

Worlds and Multiverses

- the molecules in an environment are encapsulated in a world
- alternative worlds are grouped in a multiverse

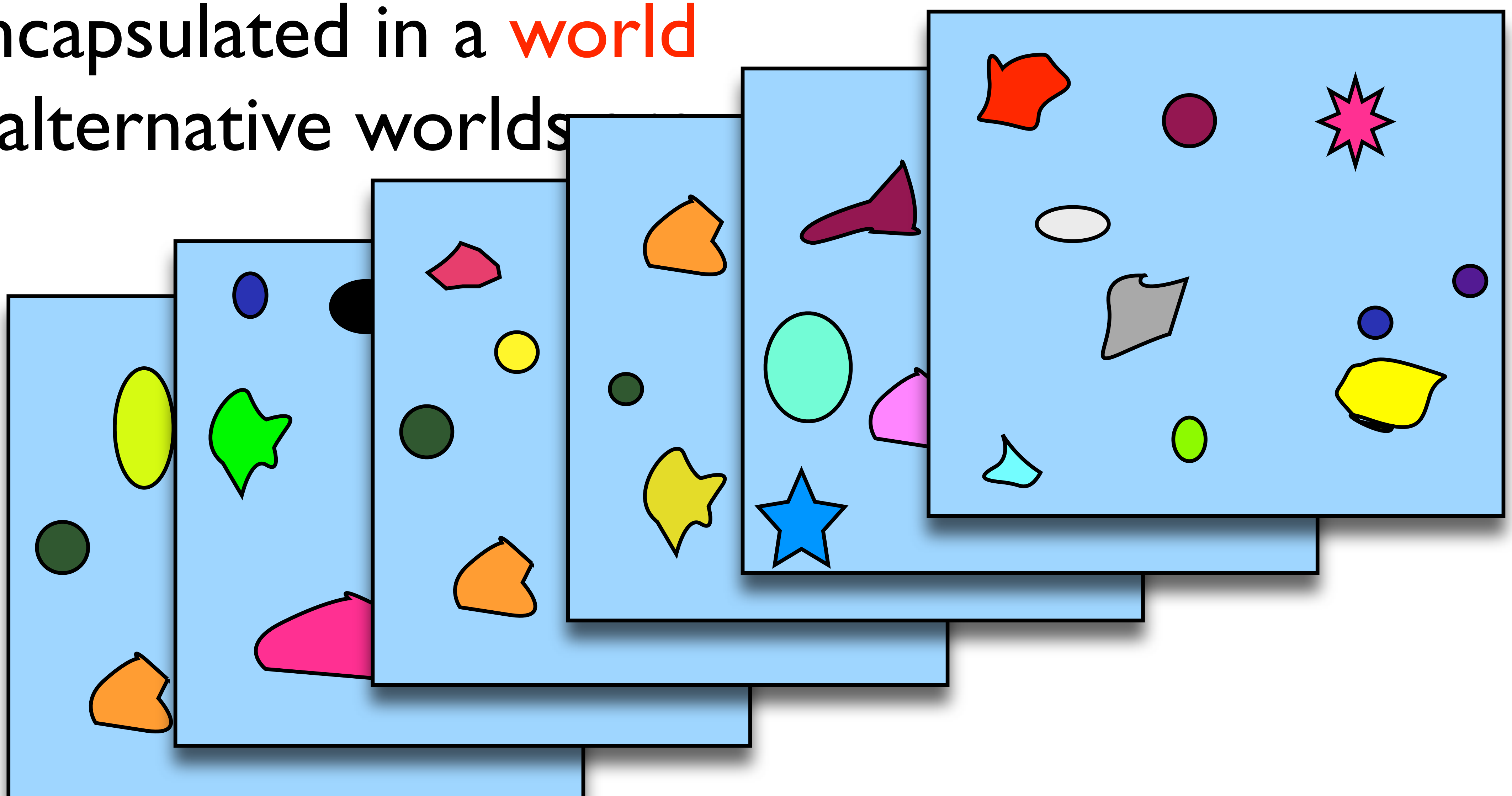


$\mid [\mathcal{M}]$

$\mid \{\mathcal{V}\ldots\mathcal{V}\} \mid \{\mathcal{W}\ldots\mathcal{W}\}$

Worlds and Multiverses

- the molecules in an environment are encapsulated in a world
- alternative worlds

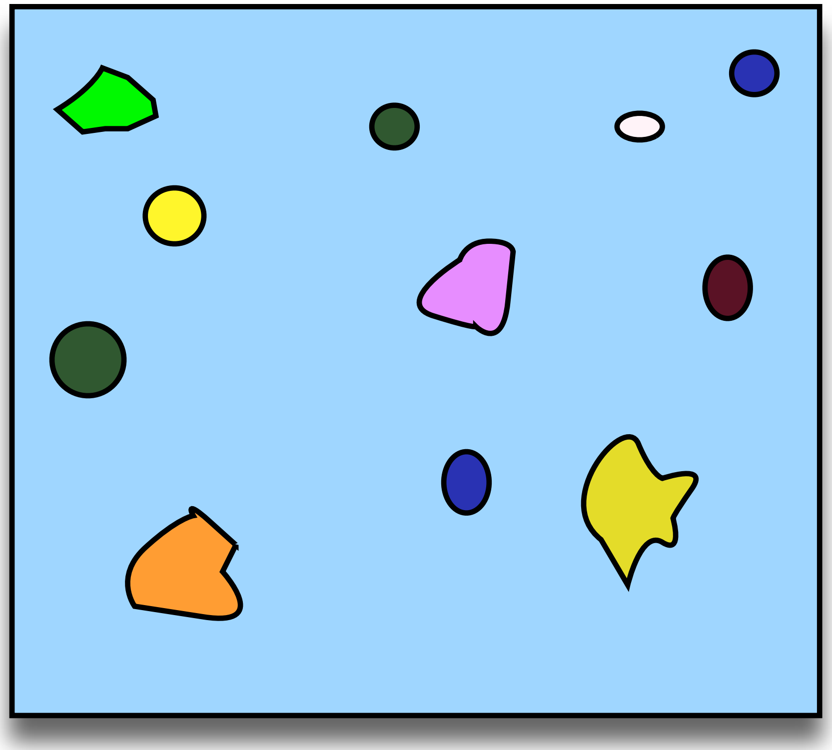


Basic Semantics

$$(L \Rightarrow R)\ M \quad \longrightarrow \quad \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\} \ \ \textbf{if} \ \ \mathcal{S}ol(L \lll M) = \{\varsigma_1, \ldots, \varsigma_n\}$$

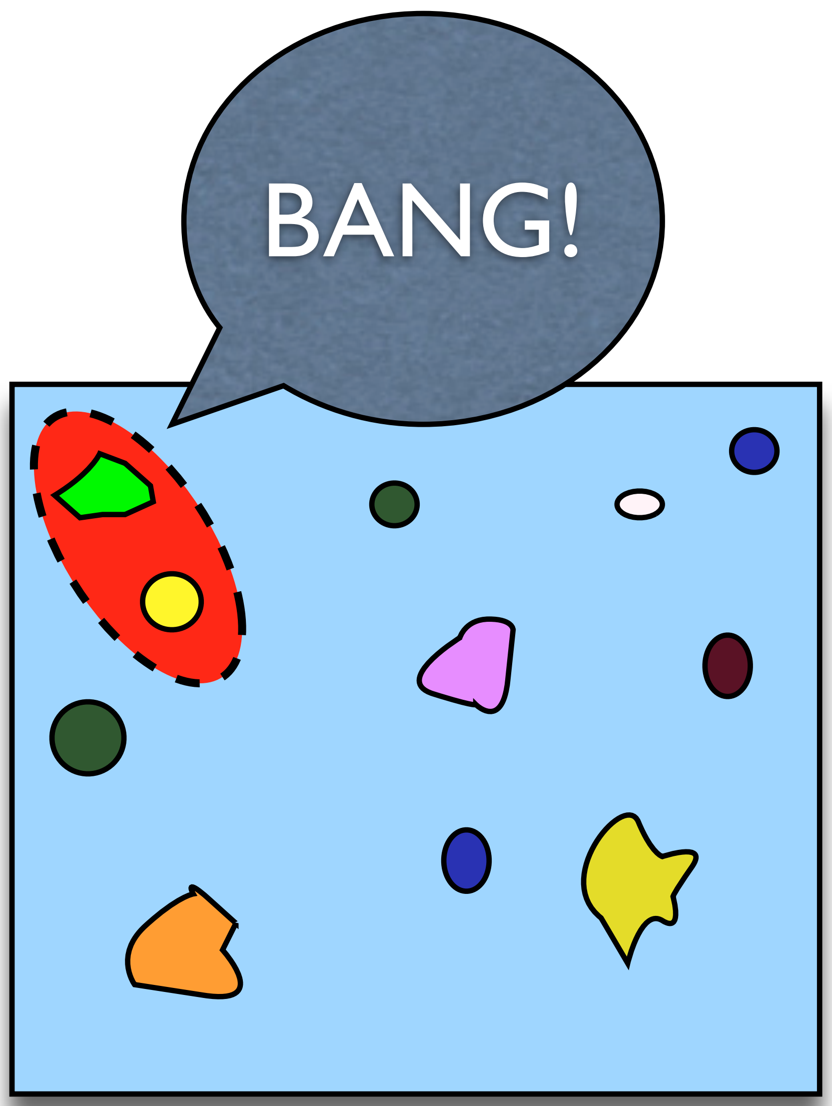
Basic Semantics

$$(L \Rightarrow R)\ M \quad \longrightarrow \quad \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\} \ \ \textbf{if} \ \ \mathcal{S}ol(L \lll M) = \{\varsigma_1, \ldots, \varsigma_n\}$$



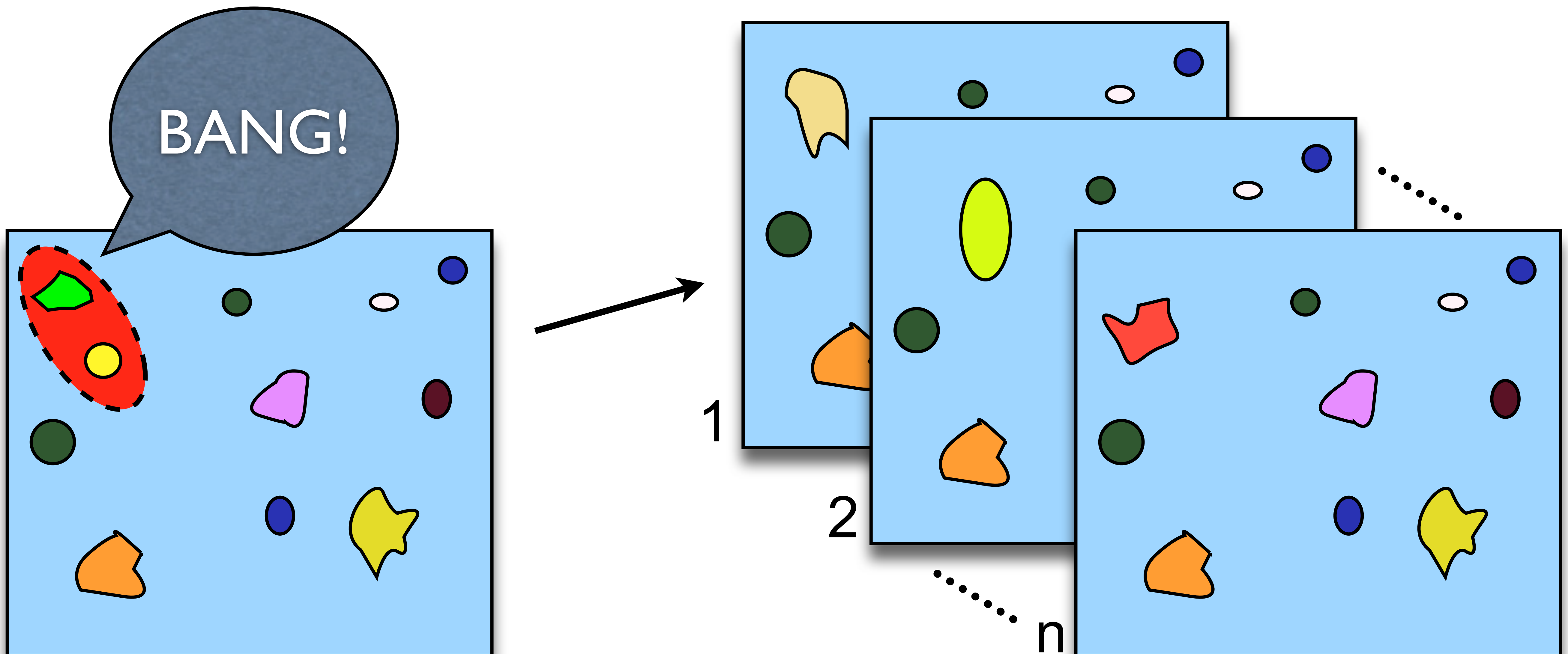
Basic Semantics

$$(L \Rightarrow R)\ M \quad \longrightarrow \quad \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\} \ \ \textbf{if} \ \ \mathcal{S}ol(L \twoheadleftarrow M) = \{\varsigma_1, \ldots, \varsigma_n\}$$



Basic Semantics

$$(L \Rightarrow R)\ M \quad \longrightarrow \quad \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\} \;\; \textbf{if} \;\; \mathcal{S}ol(L \lll M) = \{\varsigma_1, \ldots, \varsigma_n\}$$



Making the Application Explicit

(Heating) $[N\ A\ M] \longrightarrow_h [N\ A@M]$

Making the Application Explicit

$$(\textbf{Heating}) \quad [N\ A\ M] \longrightarrow_h [N\ A@M]$$

$$(\textbf{Application}) \quad (L \Rightarrow R)@M \longrightarrow_a \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\}$$

$$\textbf{if} \quad \mathcal{S}ol(L \lll M) = \{\varsigma_1, \ldots, \varsigma_n\}$$

Making the Application Explicit

(Heating) $[N\ A\ M] \longrightarrow_h [N\ A@M]$

(Application) $(L \Rightarrow R)@M \longrightarrow_a \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\}$

$\quad$ **if** $\mathcal{S}ol(L \lll M) = \{\varsigma_1, \ldots, \varsigma_n\}$

(AppFail) $(L \Rightarrow R)@M \longrightarrow_{af} (L \Rightarrow R)\ M$ **if** $\mathcal{S}ol(L \lll M) = \emptyset$

Making the Application Explicit

(Heating) $[N\ A\ M] \longrightarrow_h [N\ A@M]$

(Application) $(L \Rightarrow R)@M \longrightarrow_a \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\}$

if $\mathcal{S}ol(L \lll M) = \{\varsigma_1, \ldots, \varsigma_n\}$

(AppFail) $(L \Rightarrow R)@M \longrightarrow_{af} (L \Rightarrow R)\ M$ **if** $\mathcal{S}ol(L \lll M) = \emptyset$

(Cooling) $[N\ \{[M_1] \ldots [M_n]\}] \longrightarrow_c \{[N\ M_1] \ldots [N\ M_n]\}$

Making the Application Explicit

$$\textbf{(Heating)} \quad [N\ A\ M] \longrightarrow_h [N\ A@M]$$

$$\textbf{(Application)} \quad (L \Rightarrow R)@M \longrightarrow_a \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\}$$

$$\textbf{if} \quad \mathcal{S}ol(L \lll M) = \{\varsigma_1, \ldots, \varsigma_n\}$$

$$\textbf{(AppFail)} \quad (L \Rightarrow R)@M \longrightarrow_{af} (L \Rightarrow R)\ M \quad \textbf{if} \quad \mathcal{S}ol(L \lll M) = \emptyset$$

$$\textbf{(Cooling)} \quad [N\ \{[M_1] \ldots [M_n]\}] \longrightarrow_c \{[N\ M_1] \ldots [N\ M_n]\}$$

- introducing an explicit object for failure

Making the Application Explicit

$$\textbf{(Heating)} \quad [N\ A\ M] \longrightarrow_h [N\ A@M]$$

$$\textbf{(Application)} \quad (L \Rightarrow R)@M \longrightarrow_a \{[\varsigma_1(R)] \ldots [\varsigma_n(R)]\}$$

$$\textbf{if} \quad \mathcal{S}ol(L \lll M) = \{\varsigma_1, \ldots, \varsigma_n\}$$

$$\textbf{(AppFail)} \quad (L \Rightarrow R)@M \longrightarrow_{af} \{[\mathsf{stk}]\} \quad \textbf{if} \quad \mathcal{S}ol(L \lll M) = \emptyset$$

$$\textbf{(Cooling)} \quad [N\ \{[M_1] \ldots [M_n]\}] \longrightarrow_c \{[N\ M_1] \ldots [N\ M_n]\}$$

- introducing an explicit object for failure

Strategies

- enforce confluence and termination
- control over composing or choosing the abstractions to apply

★ strategy languages: Elan, Stratego, Tom, Maude

★ strategies: *Identity, Failure, Sequence, First, Not, IfThenElse, Try, Repeat,...*

Strategies

$$\texttt{first}(S_1, S_2) \triangleq X \Rightarrow S_1 @ X \quad (\mathsf{stk} \Rightarrow (S_2 @ X)) @ (S_1 @ X)$$

Strategies

$$\mathtt{first}(S_1, S_2) \;\triangleq\; X \Rightarrow \frac{S_1@X}{\mathsf{stk}} \quad \frac{(\mathsf{stk} \Rightarrow (S_2@X))@(\frac{S_1@X}{\mathsf{stk}})}{S_2@X}$$

Strategies

$$\texttt{first}(S_1, S_2) \quad \triangleq \quad X \Rightarrow S_1 @ X \quad (\mathsf{stk} \Rightarrow (S_2 @ X)) @ (S_1 @ X)$$

Strategies

$$\texttt{first}(S_1, S_2) \;\triangleq\; X \Rightarrow \underbrace{S_1 @ X}_{W} \;\; \underbrace{(\mathsf{stk} \Rightarrow (S_2 @ X)) @ (\underbrace{S_1 @ X}_{W})}_{\mathsf{stk}}$$

Strategy-based Extensions

- tackling application failure

$$[N\ S\ M] \longrightarrow_{hr} [N\ \ \texttt{seq}(S, \texttt{try}(\mathsf{stk} \Rightarrow S\ M))@M]$$

Strategy-based Extensions

- tackling application failure

$$[N\ S\ M] \longrightarrow_{hr} [N\ \ \mathtt{seq}(S, \mathtt{try}(\mathrm{stk} \Rightarrow S\ M))@M]$$

✦ persistent strategies $S!$

if $\quad S!@M \longrightarrow^{*} \{[S!\ M_1] \ldots [S!\ M_n]\}$

then $\quad S@M \longrightarrow^{*} \{[M_1] \ldots [M_n]\}$

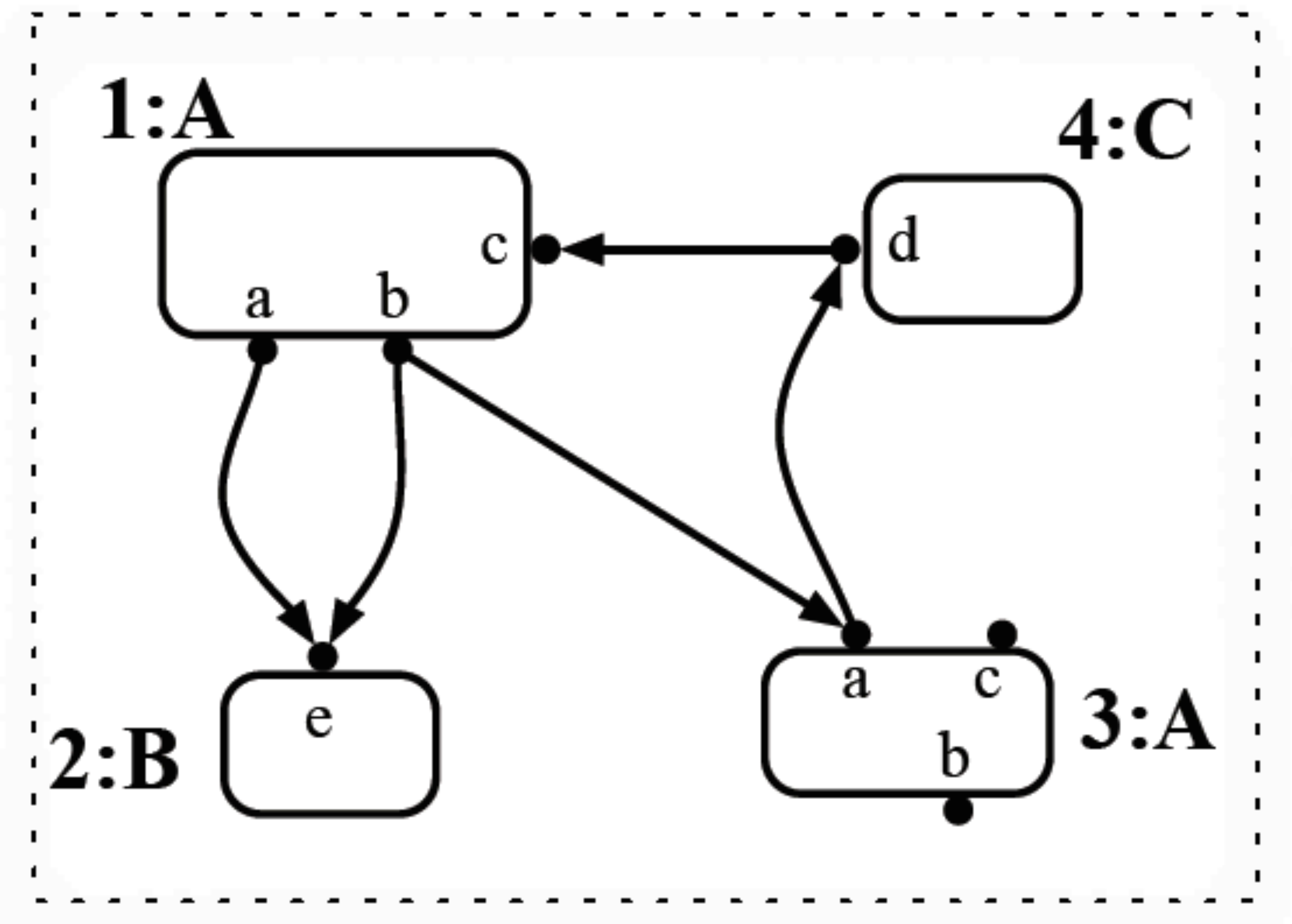
- An Abstract Biochemical Calculus
- **Port Graph Rewriting**
- A Biochemical Calculus Based on Strategic Port Graph Rewriting
- Runtime Verification in the Biochemical Calculus
- Conclusions and Perspectives

Port Graphs

- graphs with multiple edges and loops
- edges connect to ports of nodes
- defined over a signature $(\mathcal{N}, \mathcal{P})$
- category **PGraph**
  - port graphs as objects
  - node morphisms as arrows

[AndreiKirchner07-RULE]

A Port Graph

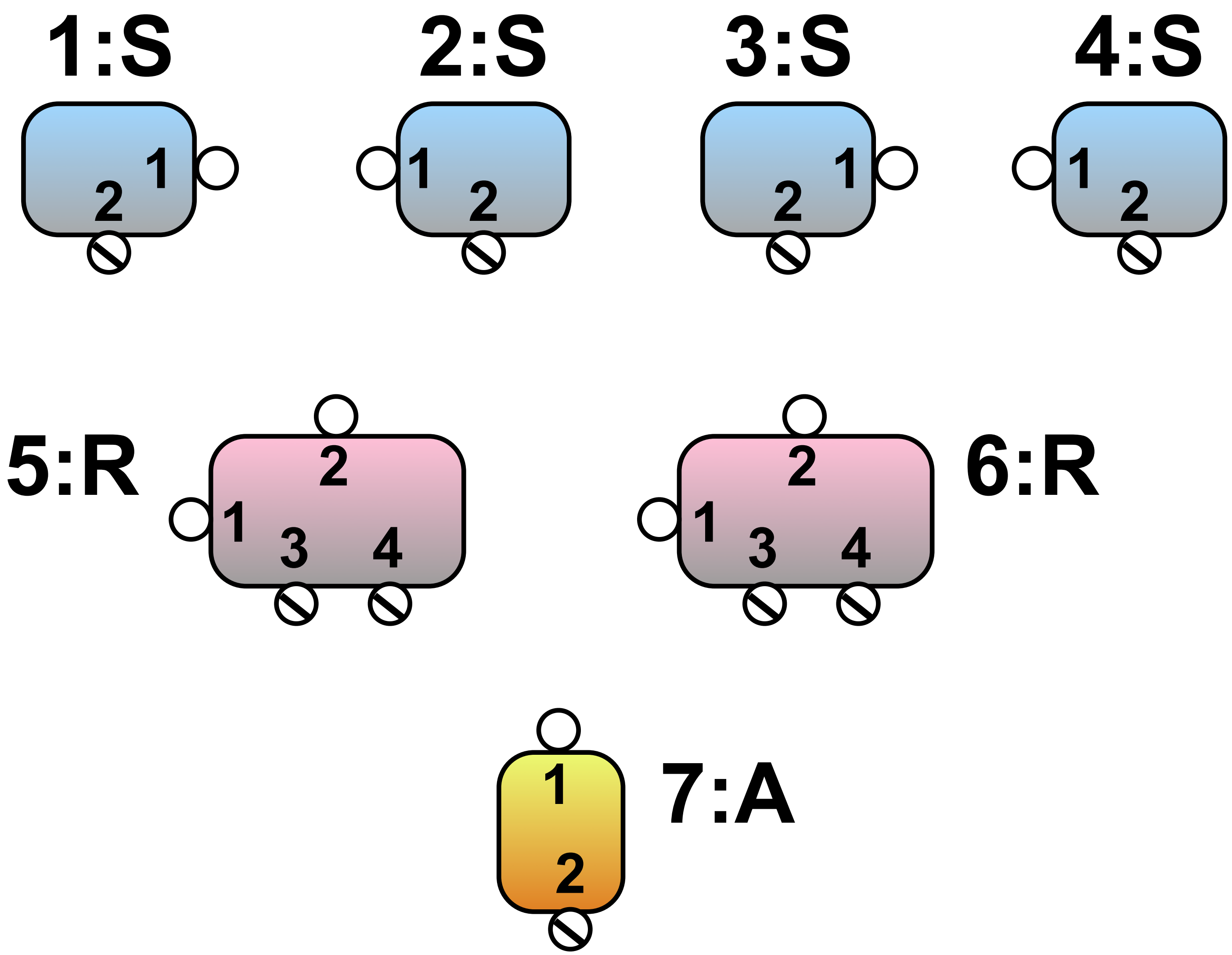


Molecular Complexes as Port Graphs

Molecular complex	Port graph
protein	node
site	port
bond	edge
interaction	rewrite rule

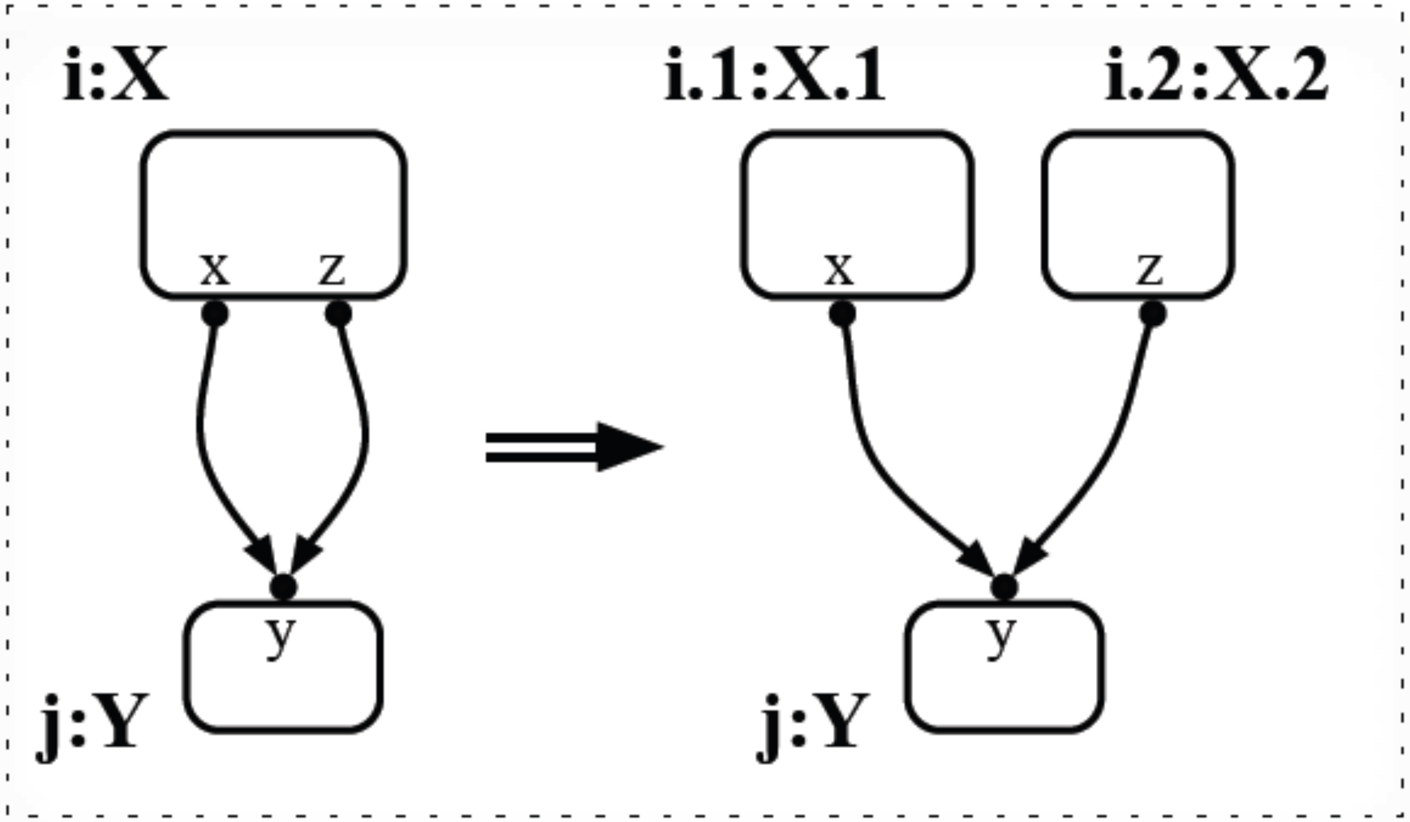
Example: a fragment of the EGFR signaling pathway

Initial state:

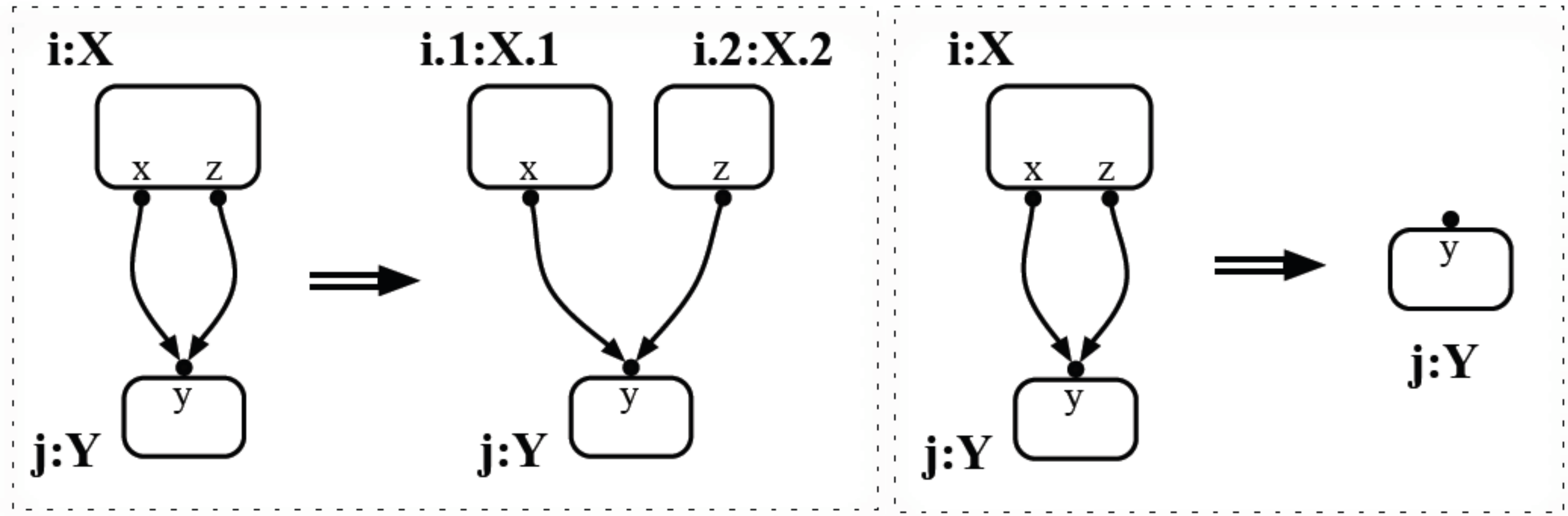


[AndreiKirchner07-NCA]

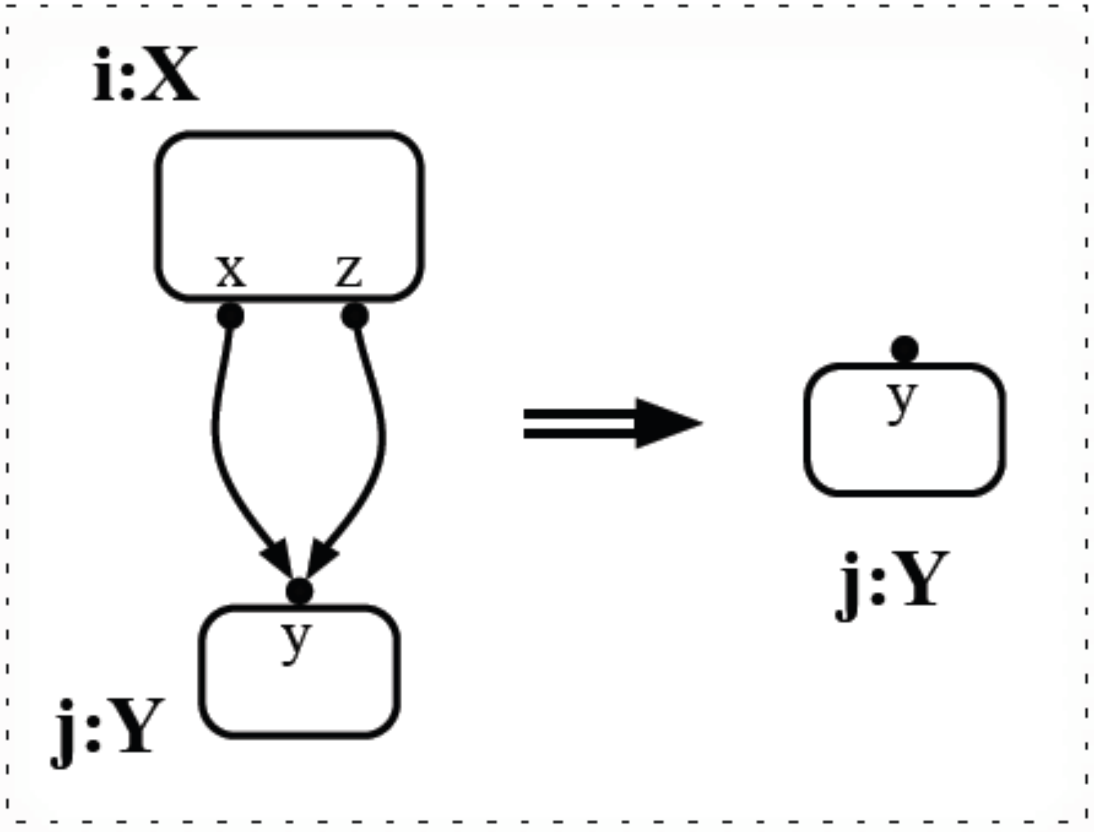
Port Graph Rewrite Rules



Port Graph Rewrite Rules

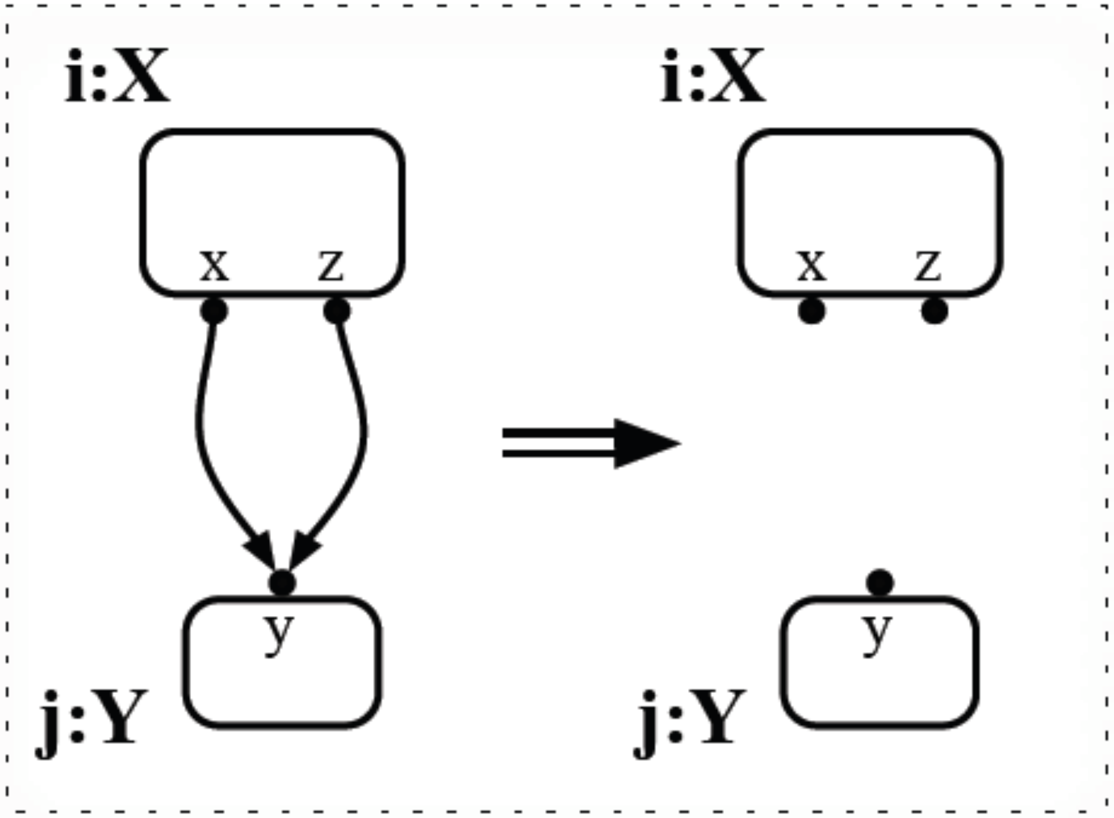


Port Graph Rewrite Rules

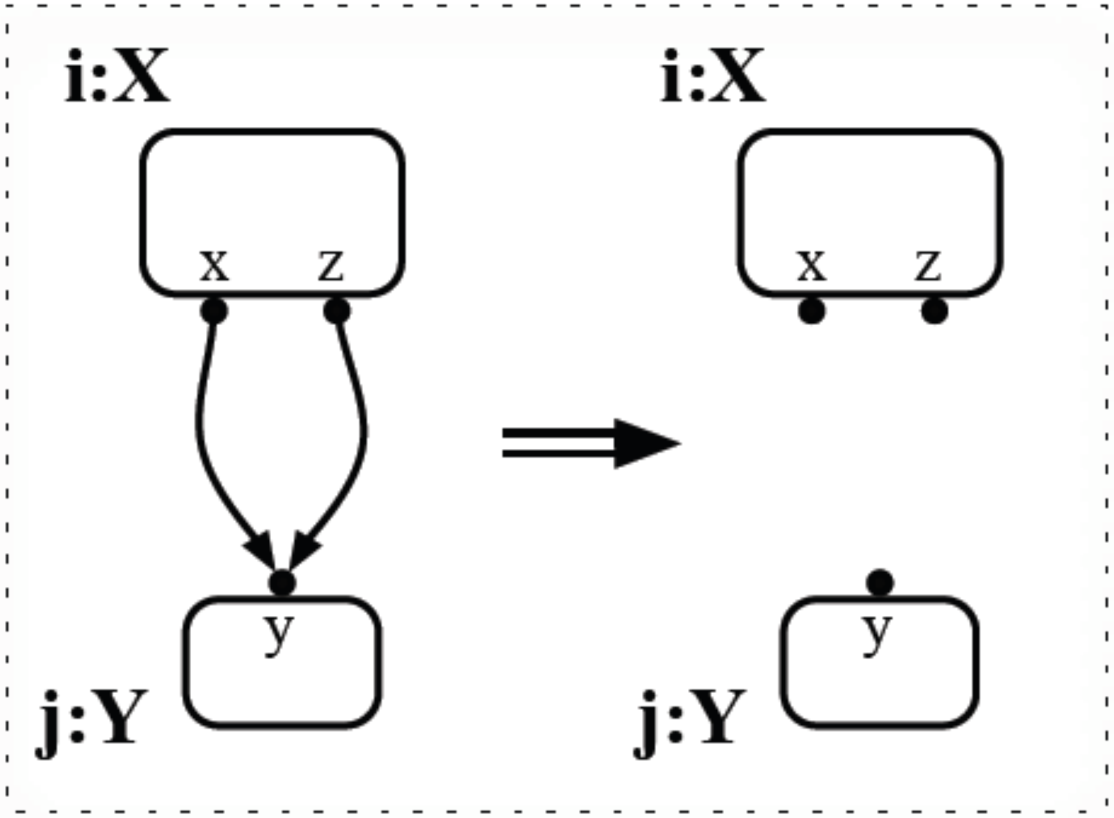


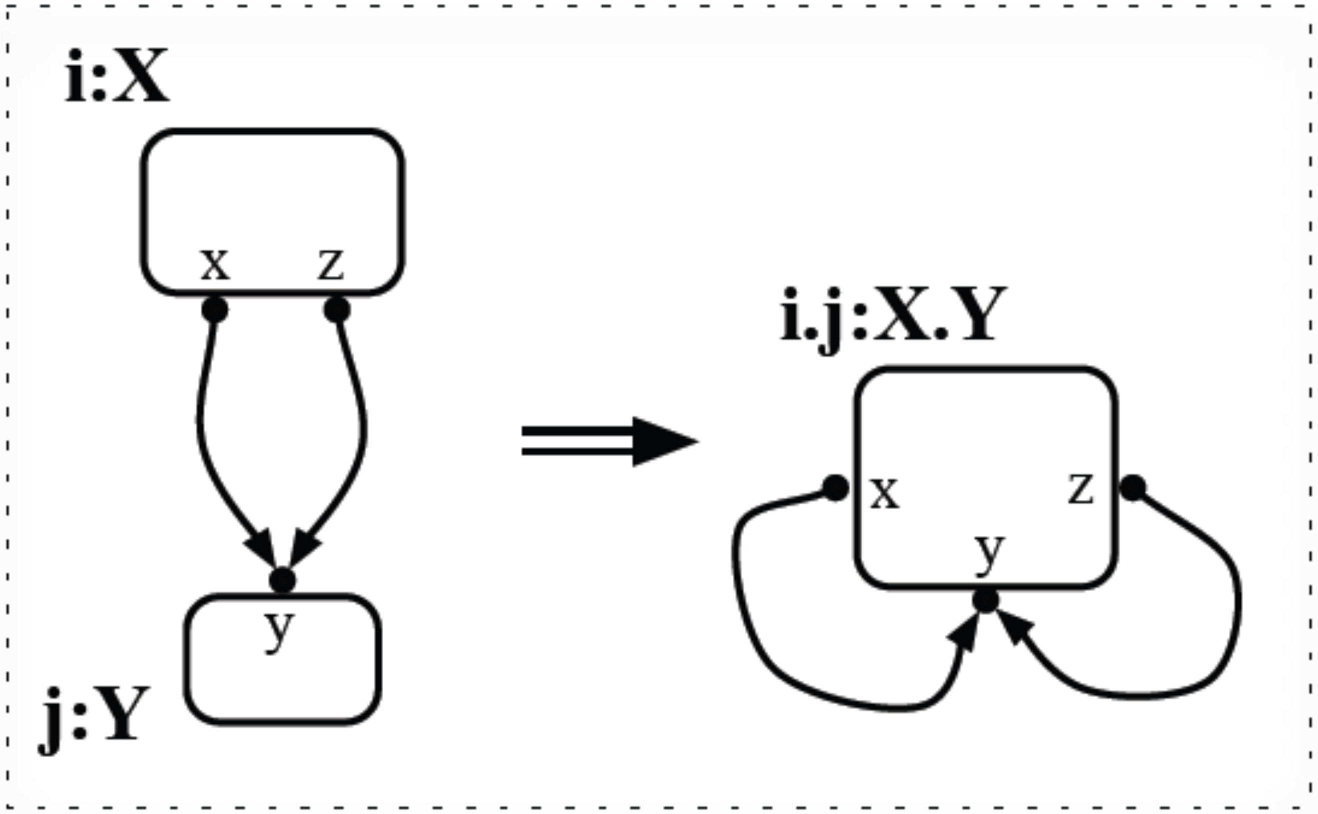
Port Graph Rewrite Rules

Port Graph Rewrite Rules

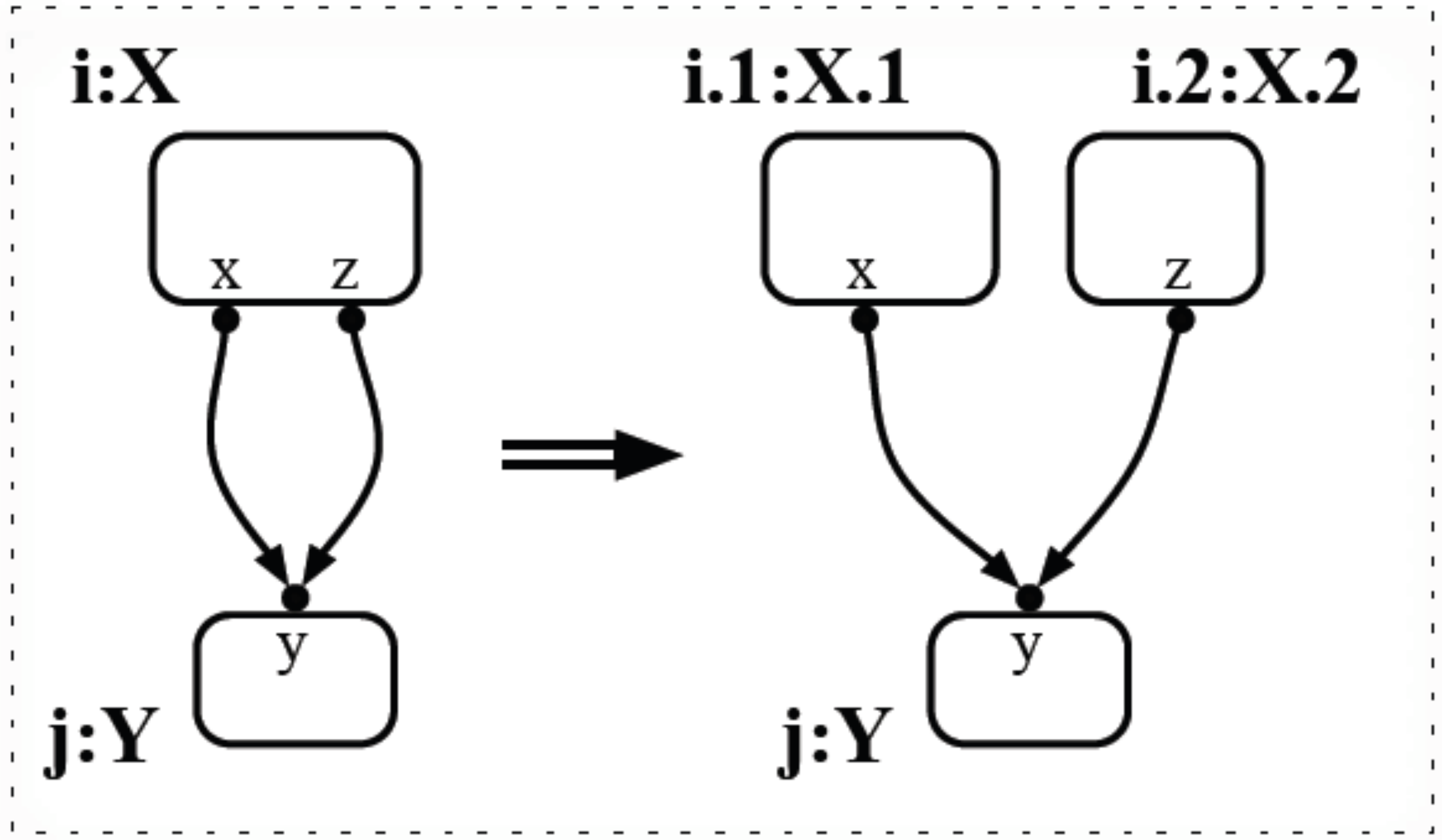


Port Graph Rewrite Rules



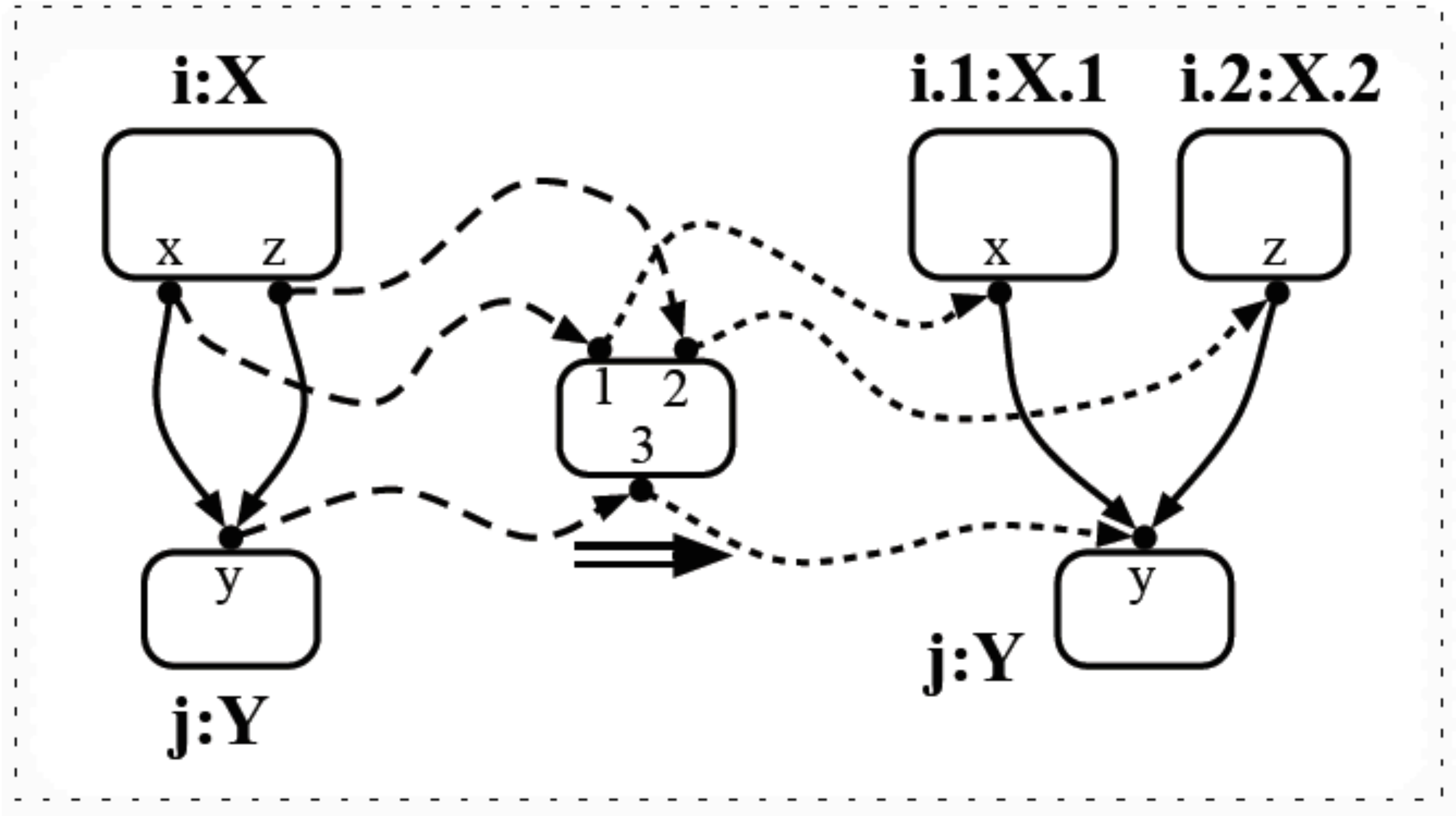


A Port Graph Rewrite Rule as a Port Graph

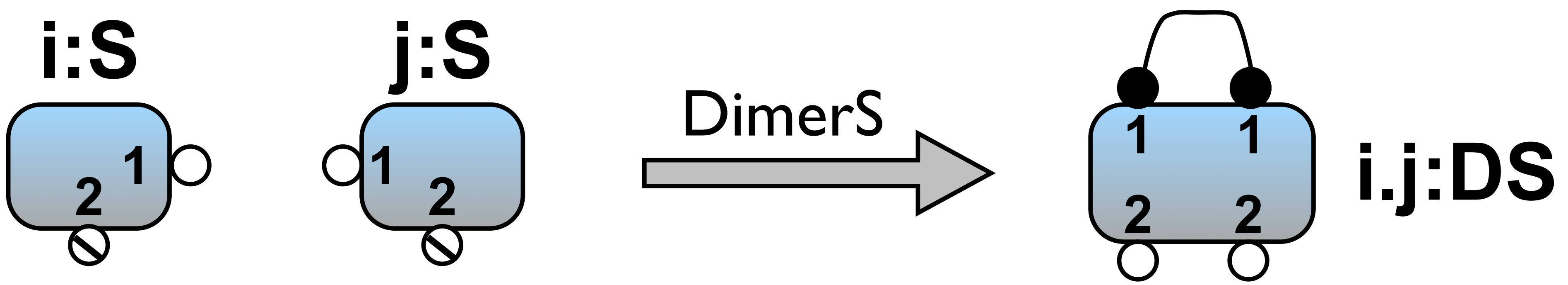


A Port Graph Rewrite Rule as a Port Graph

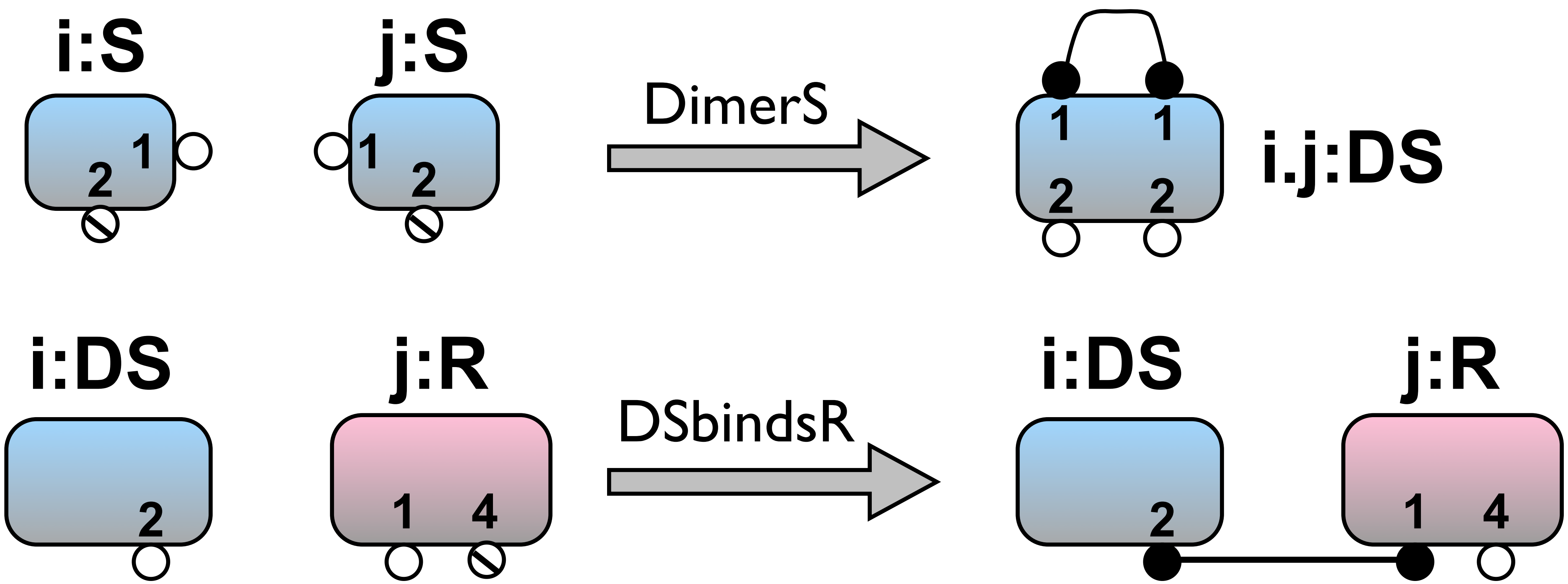
A Port Graph Rewrite Rule as a Port Graph



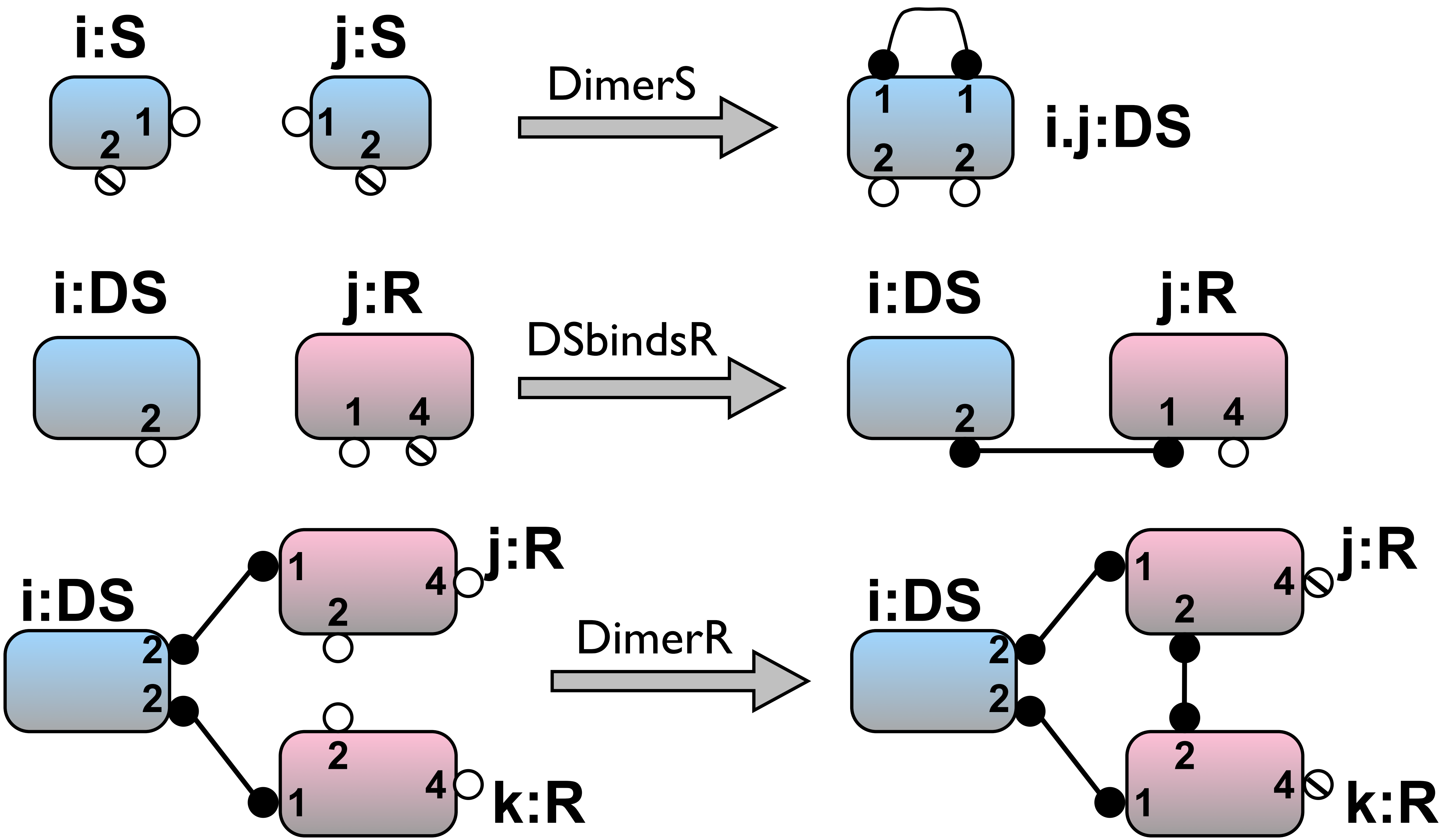
Example: a fragment of the EGFR signaling pathway



Example: a fragment of the EGFR signaling pathway



Example: a fragment of the EGFR signaling pathway



Port Graph Rewriting Relation

$$G \Rightarrow_{L \Rightarrow R} G' \quad \text{if} \quad \exists (g, G^-, \mathcal{B}) \in \mathcal{S}ol(L \lll G)$$

such that

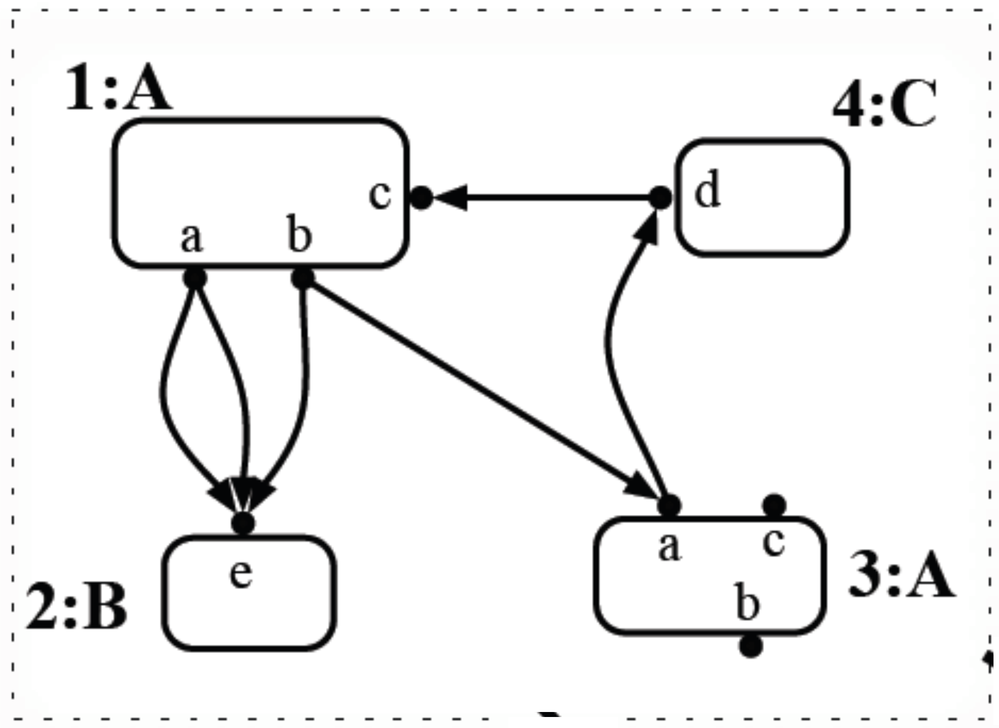
$$G = G^- \lfloor g(L) \rfloor_{\mathcal{B}}$$

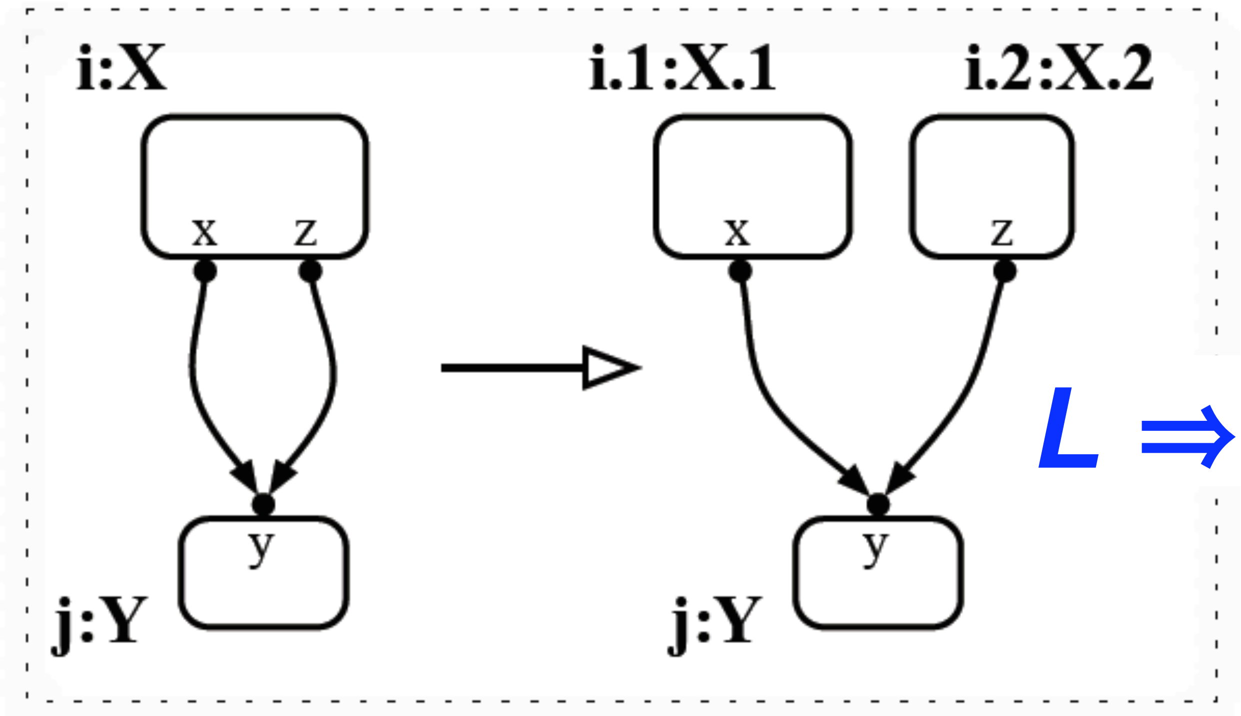
and

$$G' = G^- \lfloor g(R) \rfloor_{\Downarrow_g \mathcal{B}}$$

Port Graph Rewriting

G

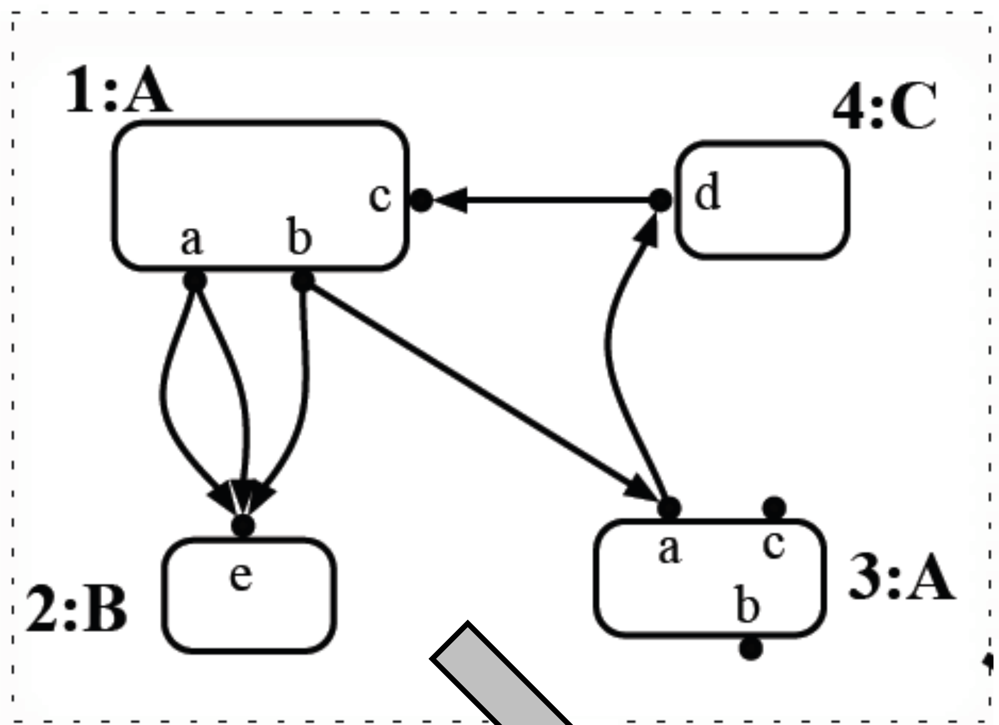


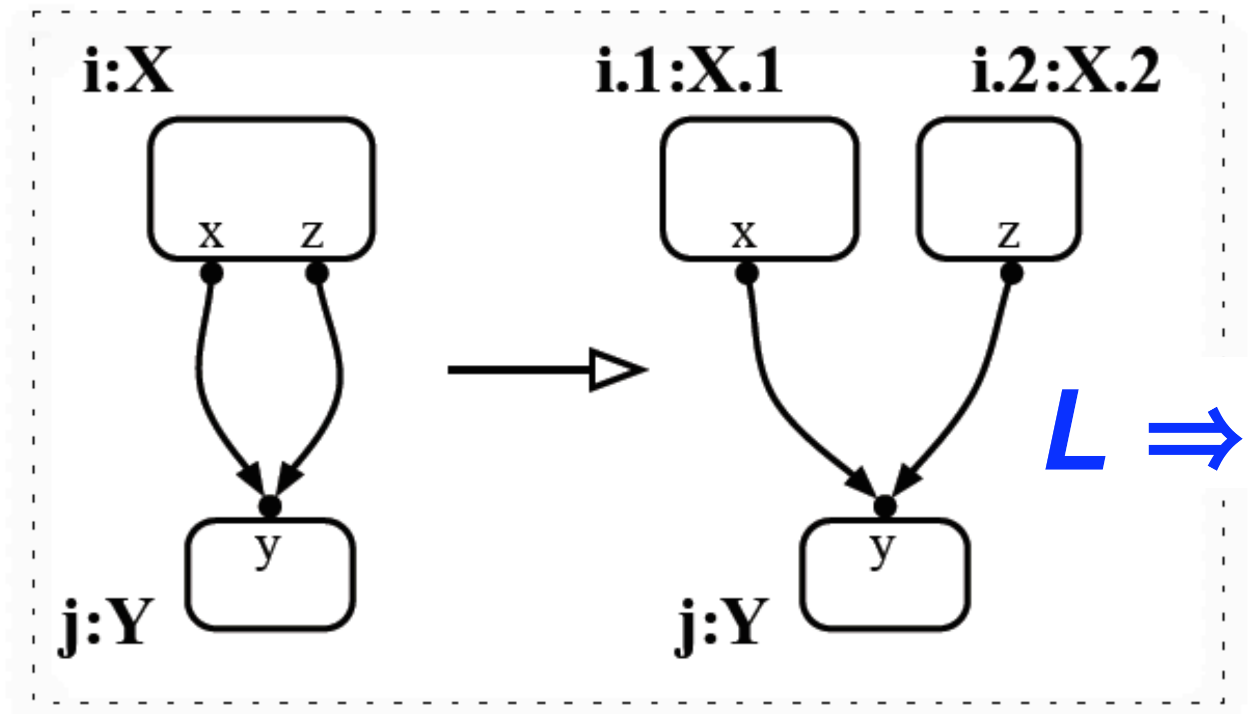


$L \Rightarrow R$

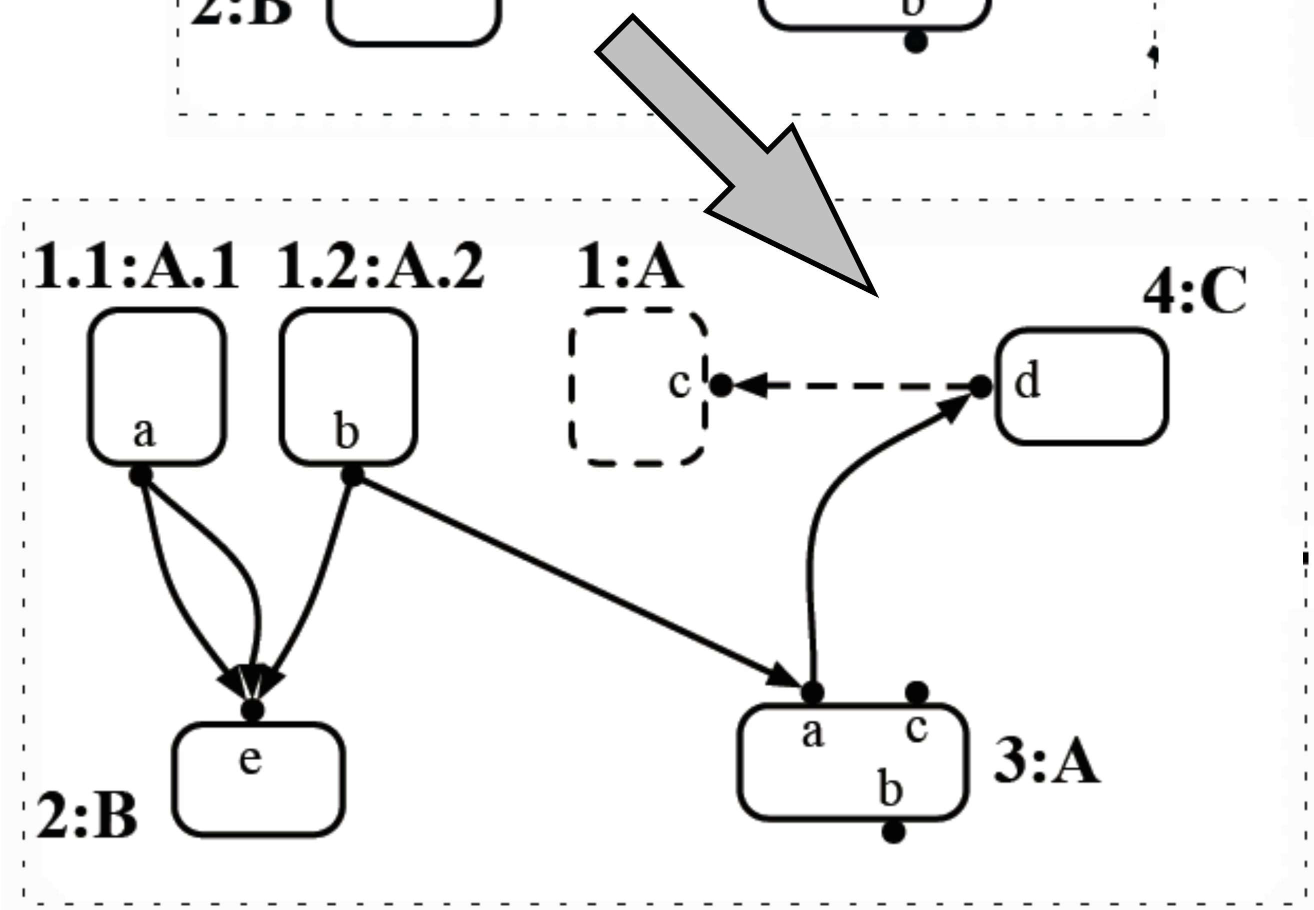
Port Graph Rewriting

G

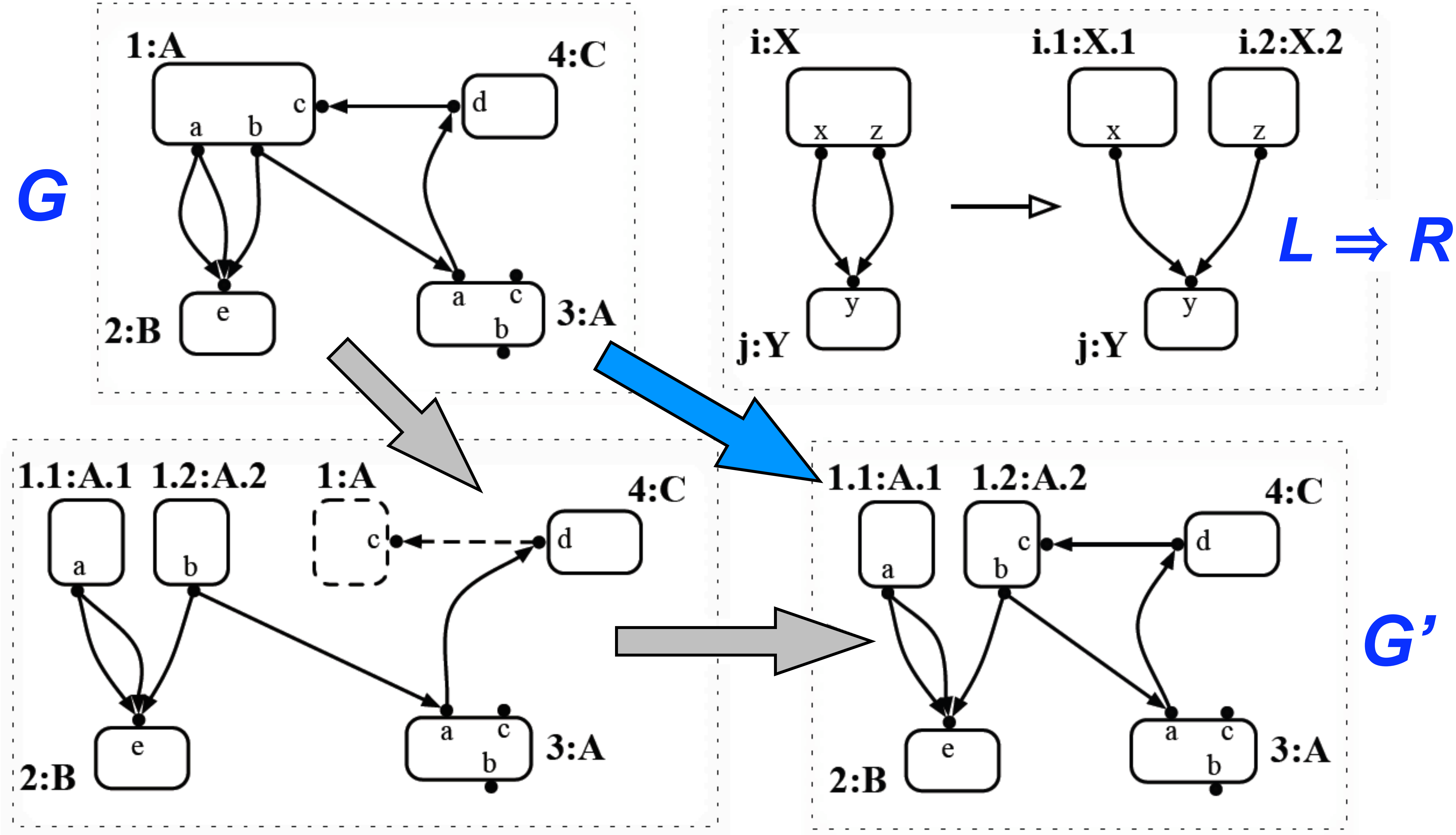




L ⇒ R

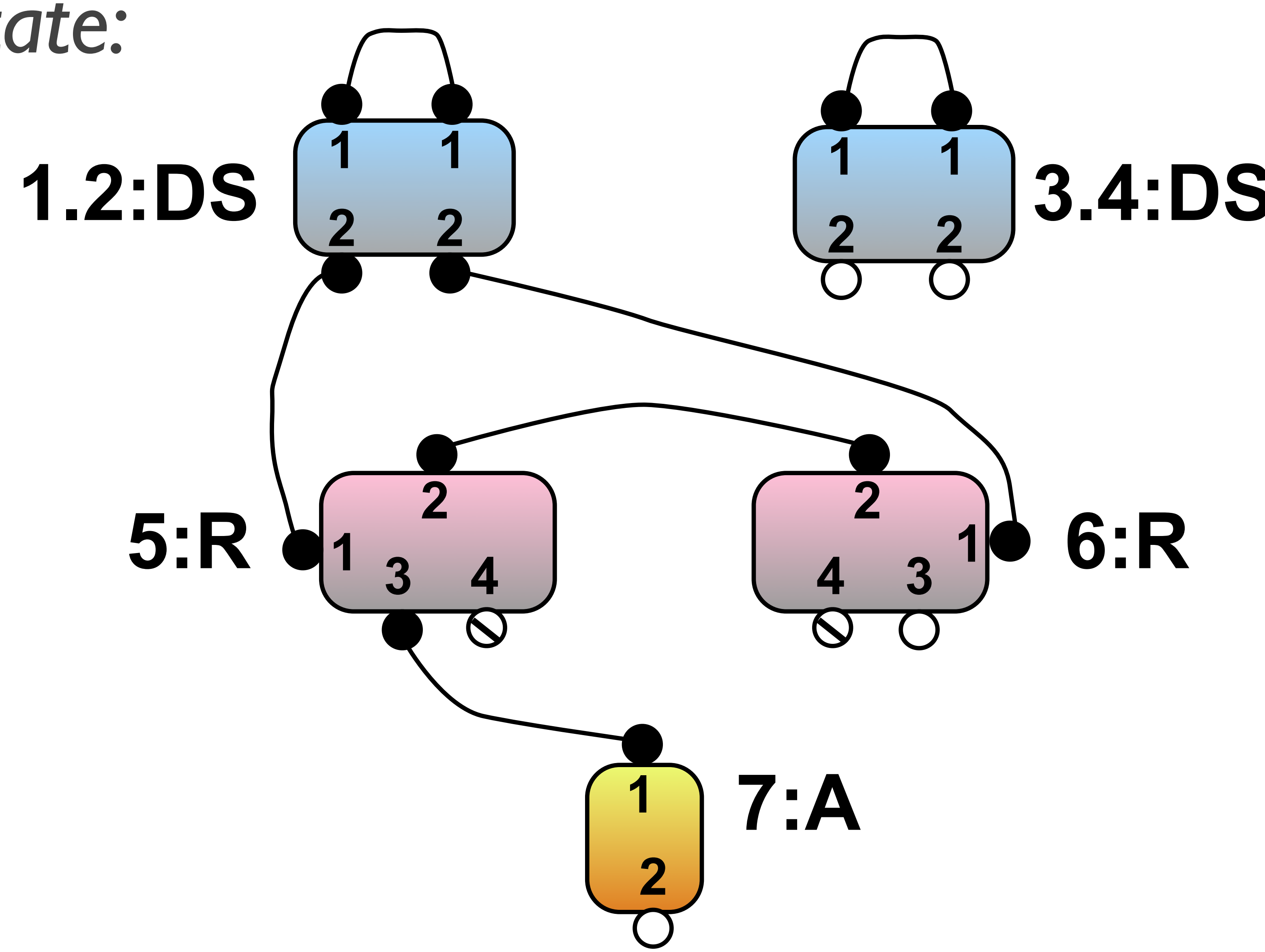


Port Graph Rewriting



Example: a fragment of the EGFR signaling pathway

A stable state:



- 2 x DimerS
- 2 x DSbindsR
- 1 x DimerR
- 2 x ActivateDR
- 1 x DRbindsA

Term Rewriting Semantics for Port Graph Rewriting

- a sound and complete axiomatization using algebraic terms
- prototype in TOM

[AndreiKirchner07-RULE]

- An Abstract Biochemical Calculus
- Port Graph Rewriting
- A Biochemical Calculus Based on Strategic Port Graph Rewriting
- Runtime Verification in the Biochemical Calculus
- Conclusions and Perspectives

A Biochemical Calculus Based on Strategic Port Graph Rewriting

-- the ρ_{Pg}-calculus --

- instantiated $\rho_{\langle\Sigma\rangle}$-calculus with the structure of port graphs
- a (sub)matching algorithm
- showed the expressivity of the port graph structure by defining the matching and replacement operations

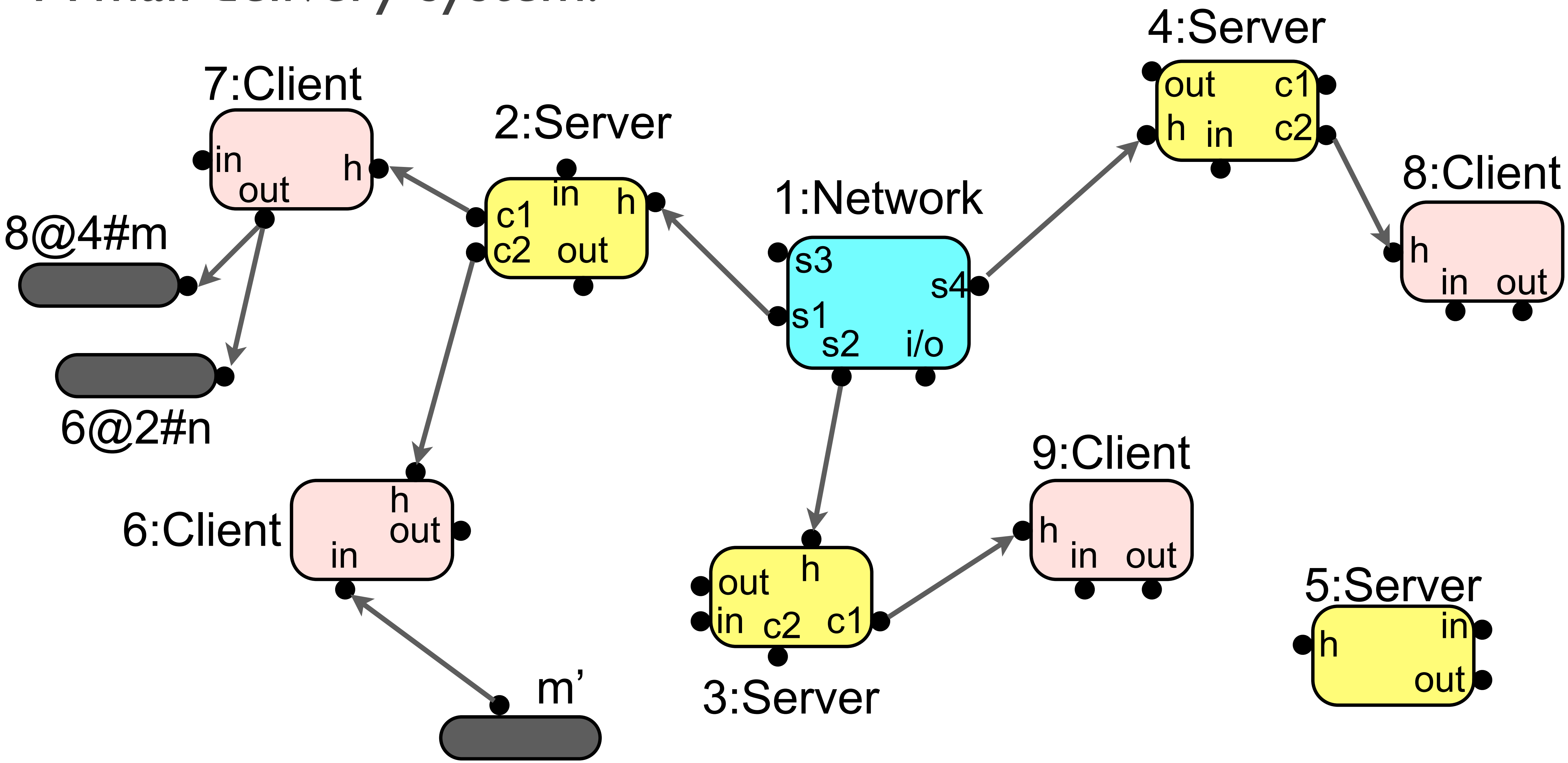
Applications: Autonomous Systems

- self-X properties:
    - self-configuration
    - self-protection
    - self-healing
    - self-optimization
- strategy-based modeling

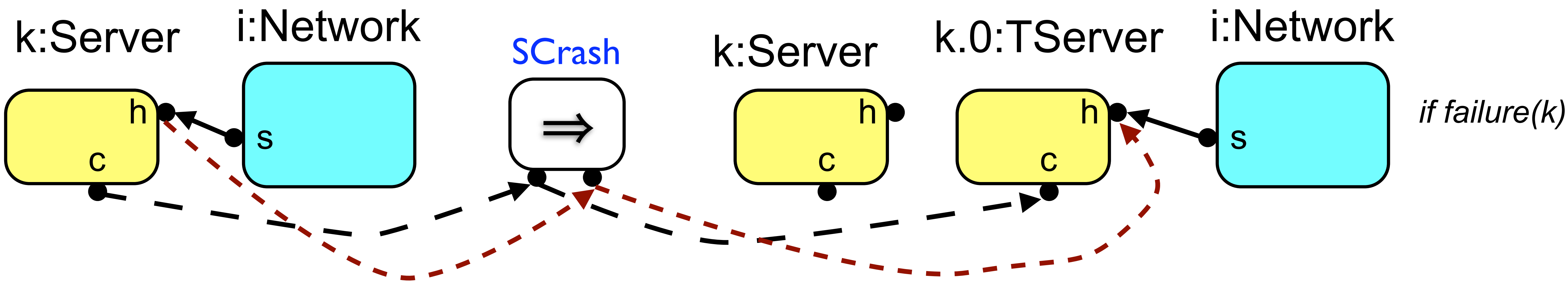
[AndreiKirchner08a,c]

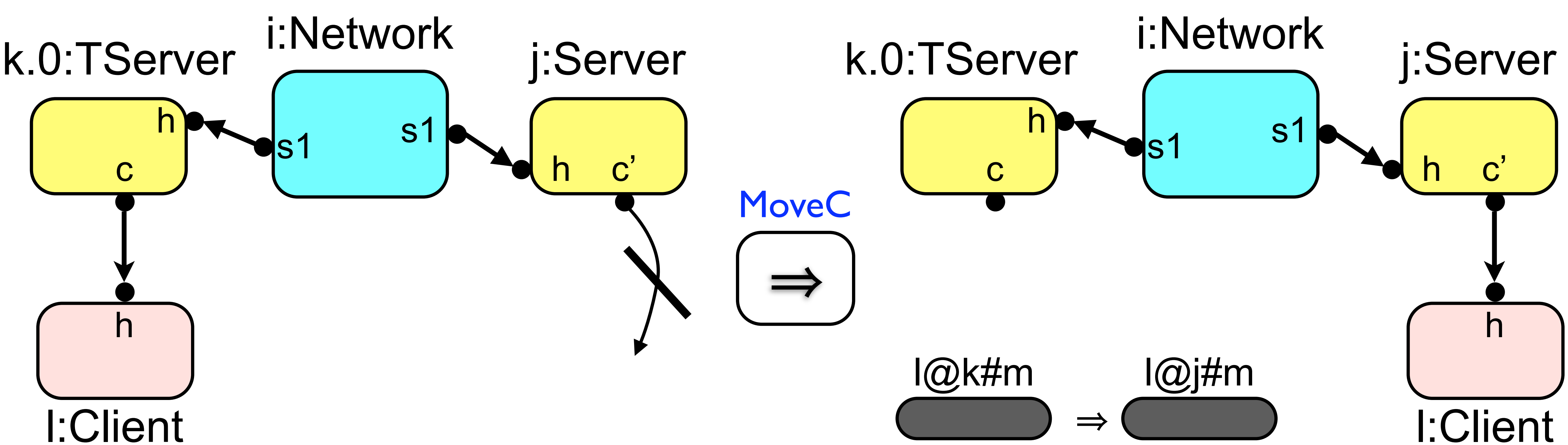
Applications: Autonomous Systems

A mail delivery system:

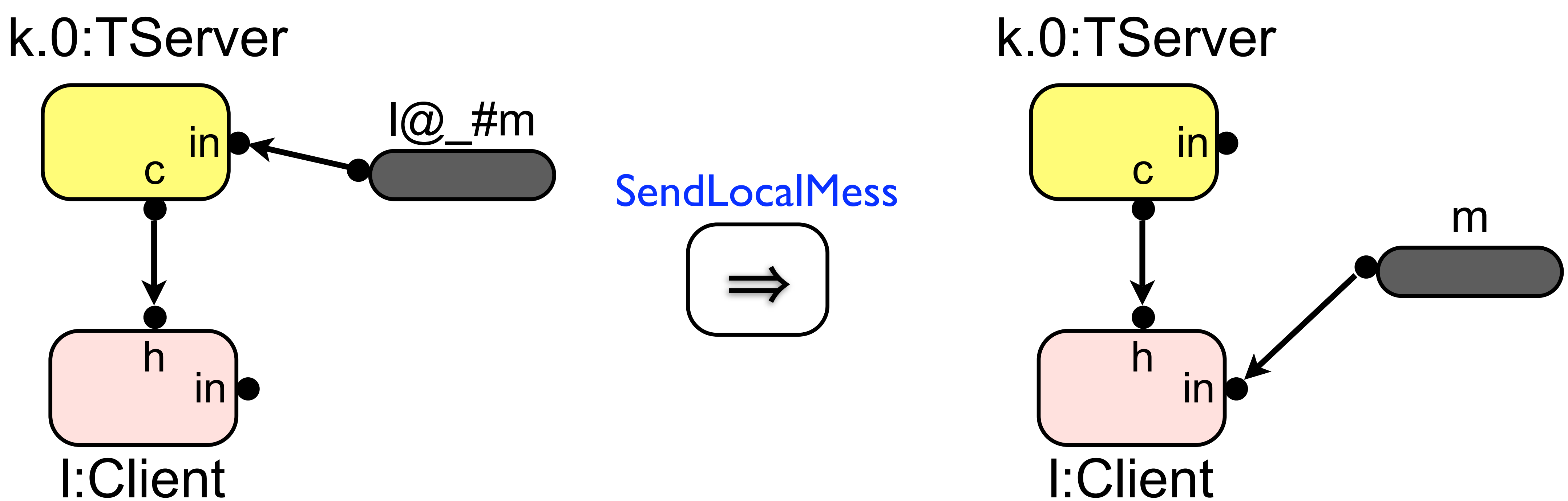


Applications: Autonomous Systems - Self-healing





Applications: Autonomous Systems - Self-healing



SCrash;

repeat(SendLocalMess);

repeat(first(MoveC, SendAwayIn, SendAwayOut))

Applications: Biochemical Networks

Initial molecular graph:

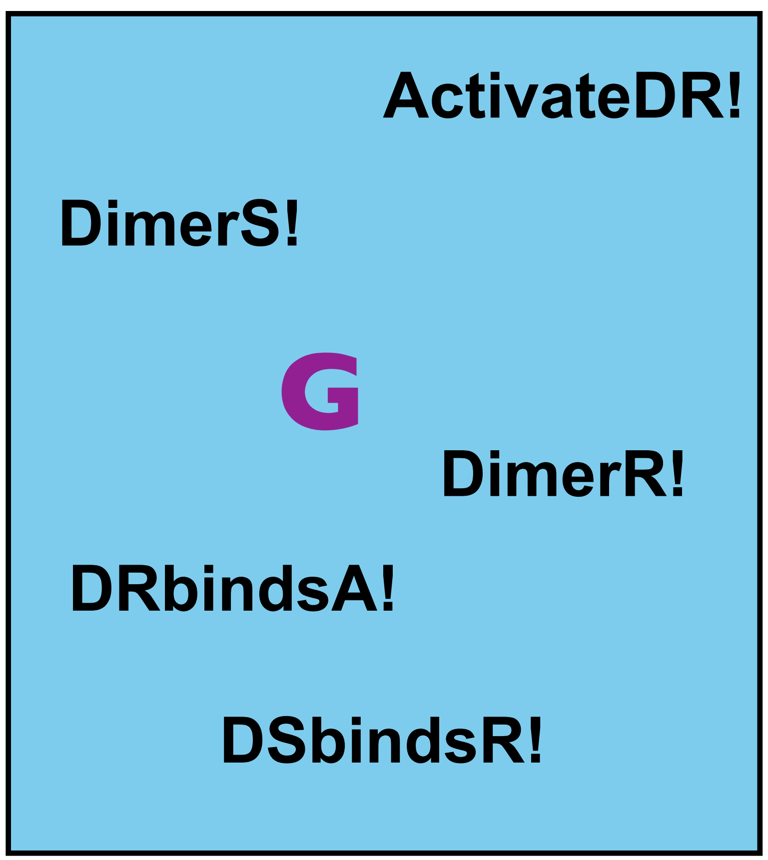
G = 1:S 2:S 3:S 4:S 5:R 6:R 7:A

Applications: Biochemical Networks

Initial molecular graph:

G = 1:S 2:S 3:S 4:S 5:R 6:R 7:A

Some possible initial states:



Applications: Biochemical Networks

Initial molecular graph:

G = 1:S 2:S 3:S 4:S 5:R 6:R 7:A

Some possible initial states:

ActivateDR!

DimerS!

G

DimerR!

DRbindsA!

DSbindsR!

ActivateDR!

`first`**(DSbindsR, DimerS)!**

G

DimerR

DRbindsA!

Applications: Biochemical Networks

Initial molecular graph:

G = 1:S 2:S 3:S 4:S 5:R 6:R 7:A

Some possible initial states:

ActivateDR!
DimerS!
G
DimerR!
DRbindsA!
DSbindsR!

ActivateDR!
`first`**(DSbindsR, DimerS)!**
G
DimerR
DRbindsA!

s
G

s = `repeat(seq(first`**(DSbindsR, DimerS), DimerR,ActivateDR, DRbindsA))**

- An Abstract Biochemical Calculus
- Port Graph Rewriting
- A Biochemical Calculus Based on Strategic Port Graph Rewriting
- **Runtime Verification in the Biochemical Calculus**
- Conclusions and Perspectives

Motivation

- invariant:
  - rule $G \Rightarrow G$
  - strategy `first`$(G \Rightarrow G, X \Rightarrow$ "*Failure*"$)!$
- remove $(G \Rightarrow$ "*Failure*"$)!$ or "repair" $(G \Rightarrow H)!$
- more: a modal logic with structural and temporal formulas → $\rho^{\mathrm{v}}{}_{\mathrm{pg}}$-calculus

Structural Formulas

Structural Formulas

Structural formulas:

$$\varphi ::= \top \mid \bot \mid \gamma \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \lozenge\varphi$$

Structural Formulas

Structural formulas:

$$\varphi ::= \top \mid \bot \mid \gamma \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \diamondsuit\varphi$$

Satisfaction relation:

$$\begin{array}{lcl} G \models \gamma & \Leftrightarrow & \exists\sigma \text{ such that } G = \sigma(\gamma) \\ G \models \diamondsuit\varphi & \Leftrightarrow & \exists G' \sqsubseteq G \text{ such that } G' \models \varphi \end{array}$$

Mapping Structural Formulas to Strategies

$$
\begin{array}{lcl}
\tau(\top) & = & \mathsf{id} \\
\tau(\bot) & = & \mathtt{fail} \\
\tau(\diamondsuit\gamma) & = & \gamma \Rightarrow \gamma \\
\tau(\neg\varphi) & = & \mathtt{not}(\tau(\varphi)) \\
\tau(\varphi_1 \wedge \varphi_2) & = & \mathtt{seq}(\tau(\varphi_1), \tau(\varphi_2)) \\
\tau(\varphi_1 \vee \varphi_2) & = & \mathtt{first}(\tau(\varphi_1), \tau(\varphi_2)) \\
\tau(\varphi_1 \rightarrow \varphi_2) & = & X \Rightarrow \mathtt{seq}(\tau(\varphi_1), \mathtt{first}(\mathsf{stk} \Rightarrow X, \tau(\varphi_2)))\mathtt{@}X
\end{array}
$$

Mapping Structural Formulas to Strategies

$$
\begin{array}{lcl}
\tau(\top) & = & \mathsf{id} \\
\tau(\bot) & = & \mathtt{fail} \\
\tau(\Diamond\gamma) & = & \gamma \Rightarrow \gamma \\
\tau(\neg\varphi) & = & \mathtt{not}(\tau(\varphi)) \\
\tau(\varphi_1 \wedge \varphi_2) & = & \mathtt{seq}(\tau(\varphi_1), \tau(\varphi_2)) \\
\tau(\varphi_1 \vee \varphi_2) & = & \mathtt{first}(\tau(\varphi_1), \tau(\varphi_2)) \\
\tau(\varphi_1 \rightarrow \varphi_2) & = & X \Rightarrow \mathtt{seq}(\tau(\varphi_1), \mathtt{first}(\mathsf{stk} \Rightarrow X, \tau(\varphi_2)))@X
\end{array}
$$

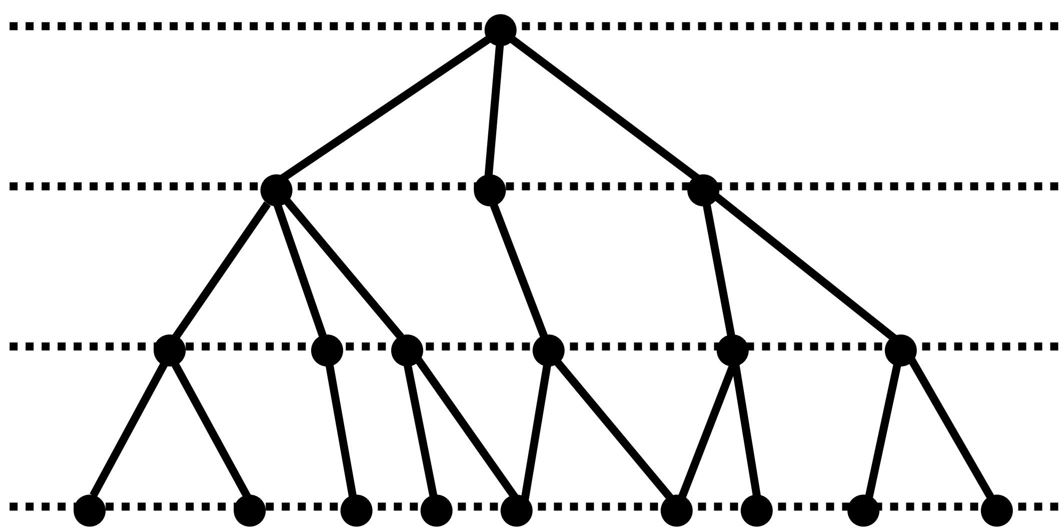
$G \not\models \varphi$ if and only if $\tau(\varphi)@G \longrightarrow^* \{[\mathsf{stk}]\}$

$G \models \varphi$ if and only if $\tau(\varphi)@G \longrightarrow^* \{[G]\}$

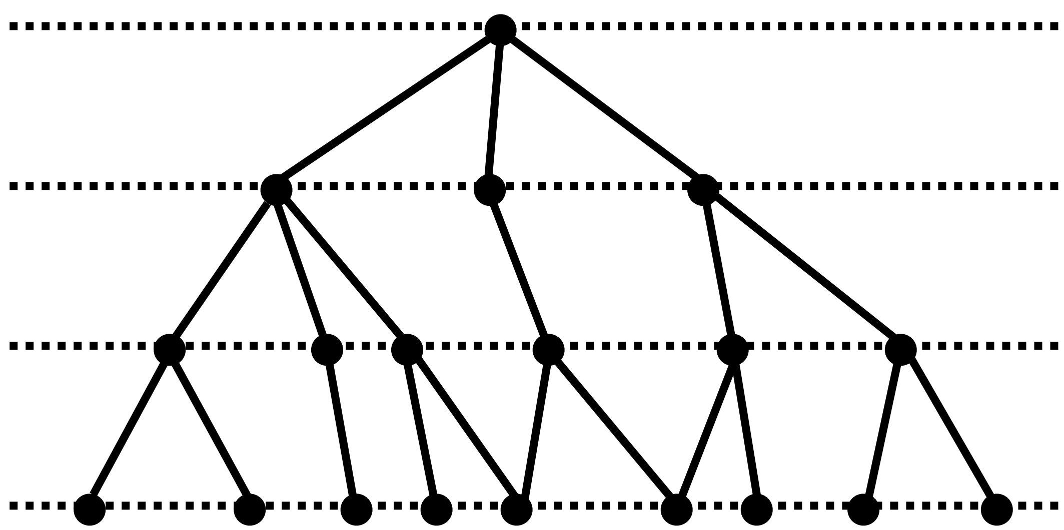
Temporal Formulas

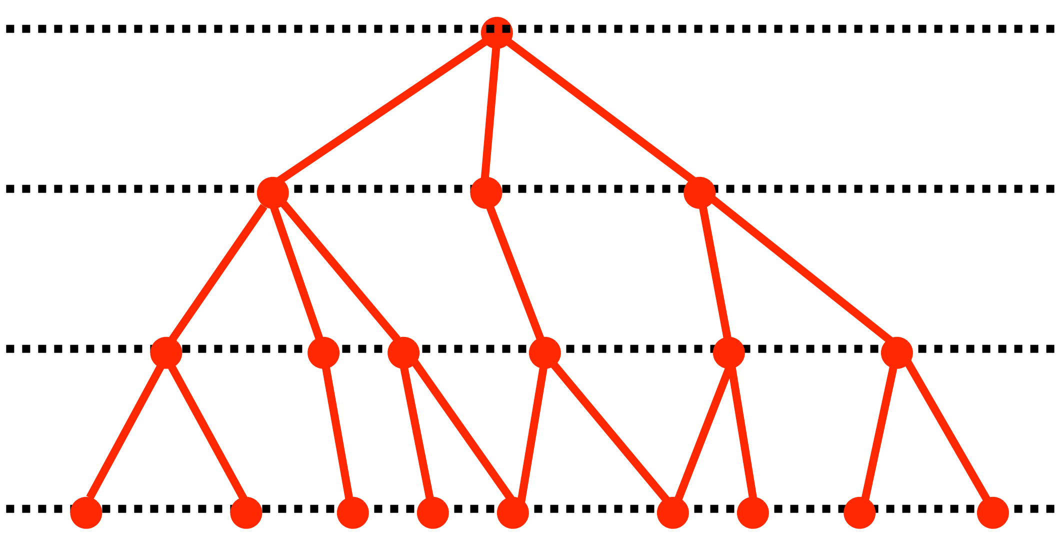
- a subset of CTL formulas constructed using:
  - a path quantifier -- A, E
  - a temporal operator -- X, G, F, U
  - structural formula(s) φ

Temporal Formulas



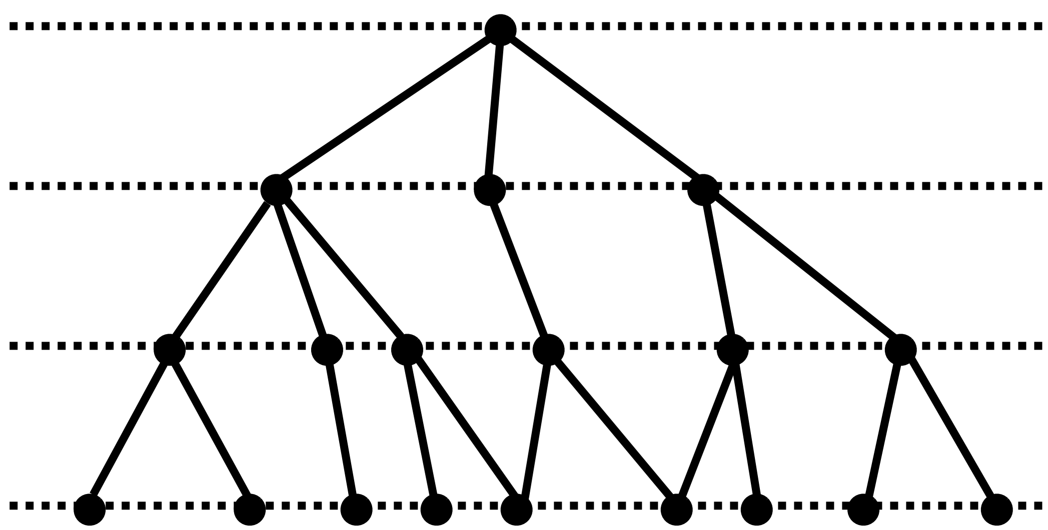
Temporal Formulas

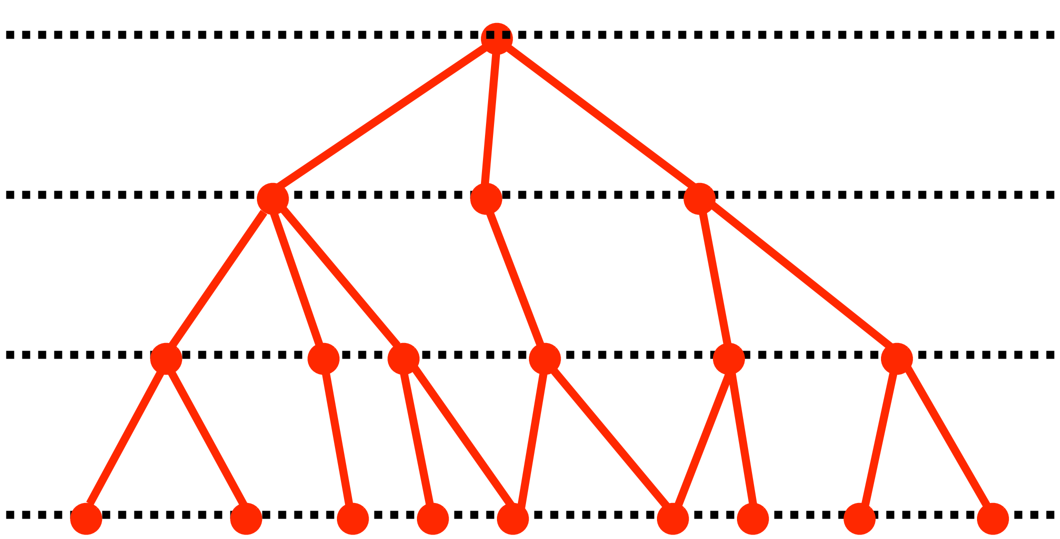




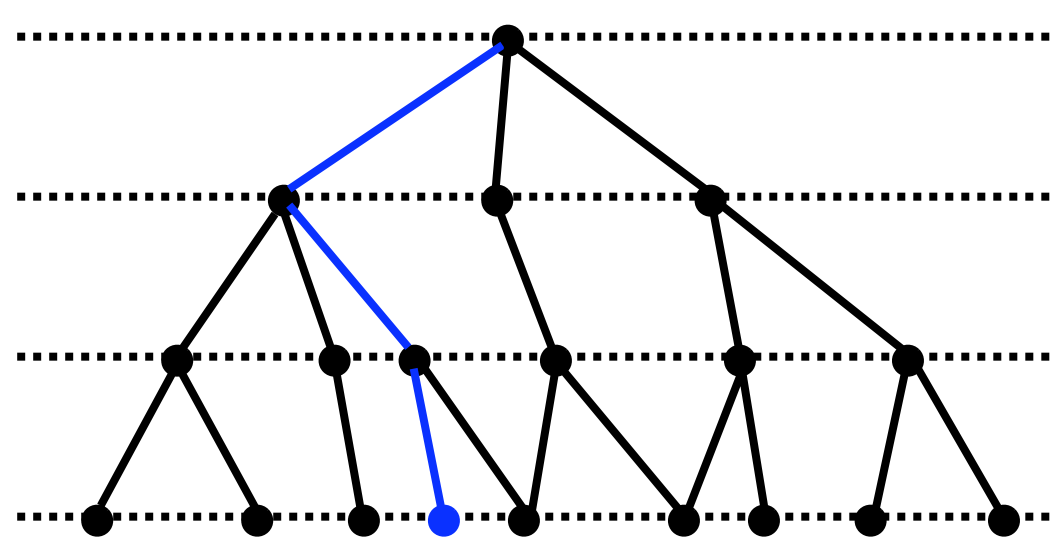
AGφ

Temporal Formulas



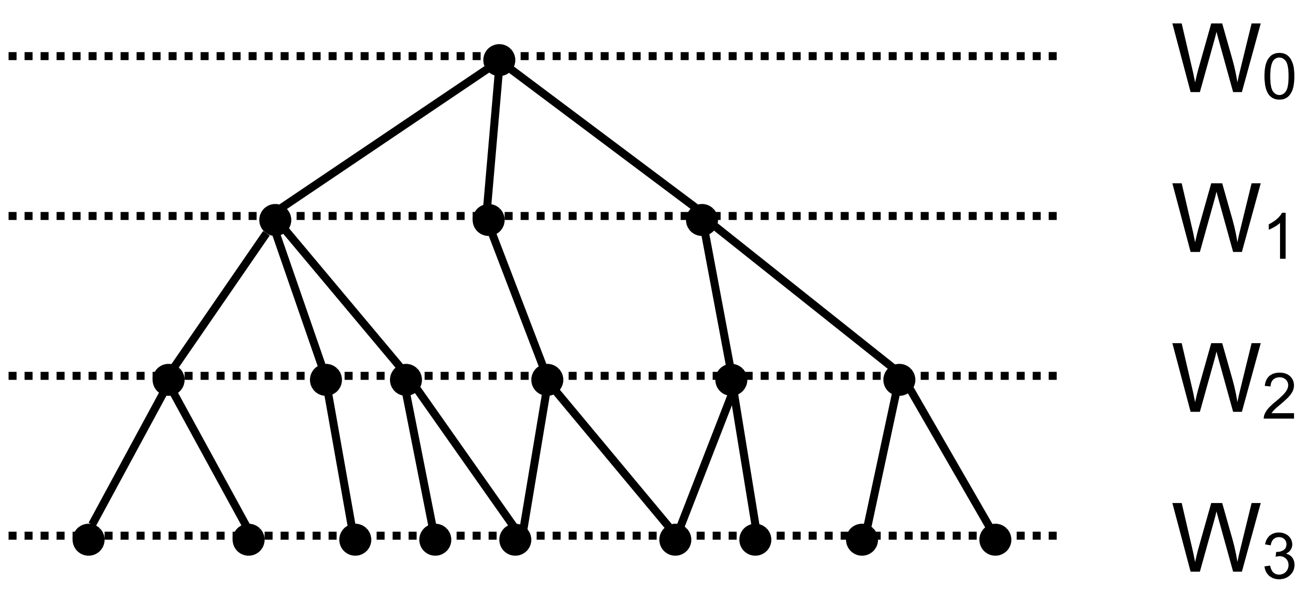


AGφ

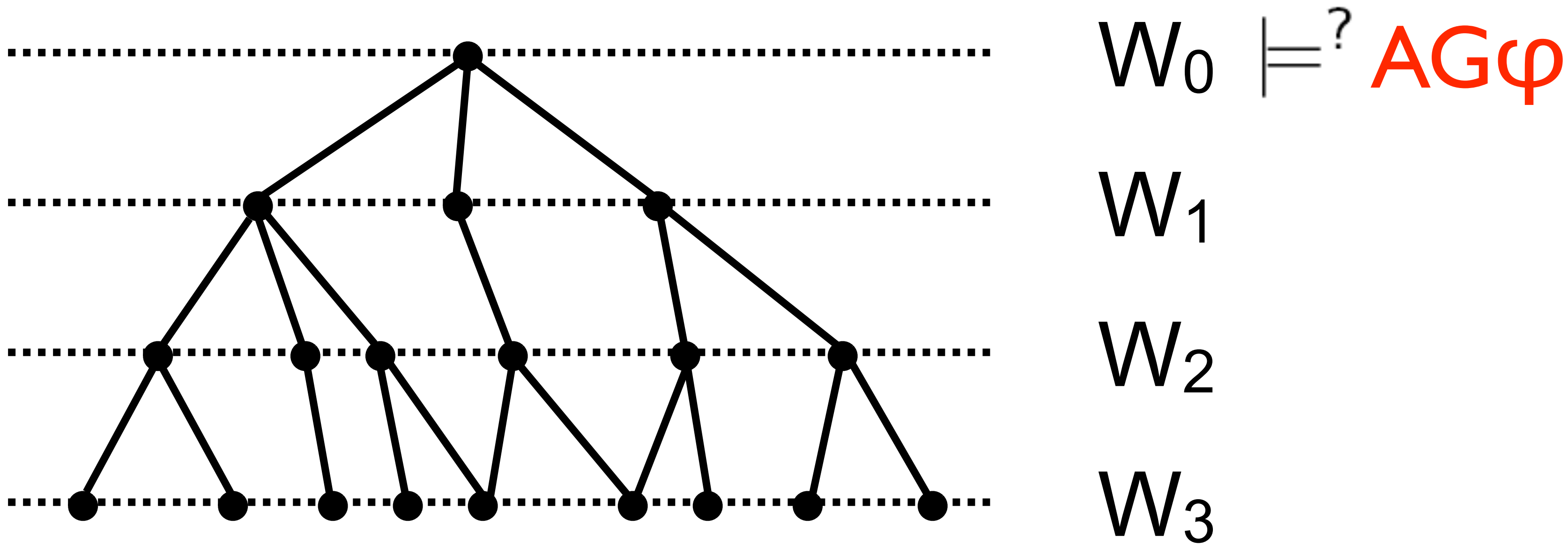


EFφ

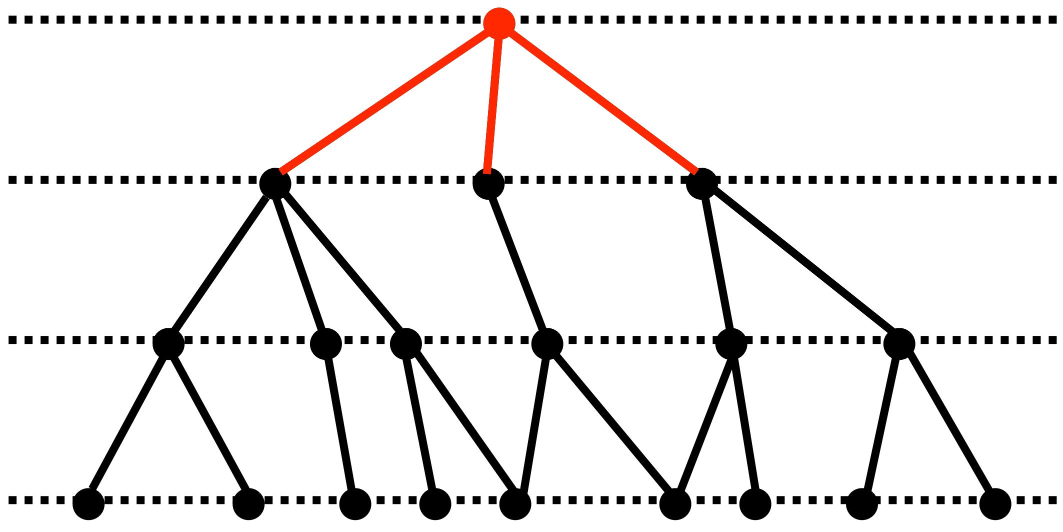
Runtime Verification



Runtime Verification



Runtime Verification



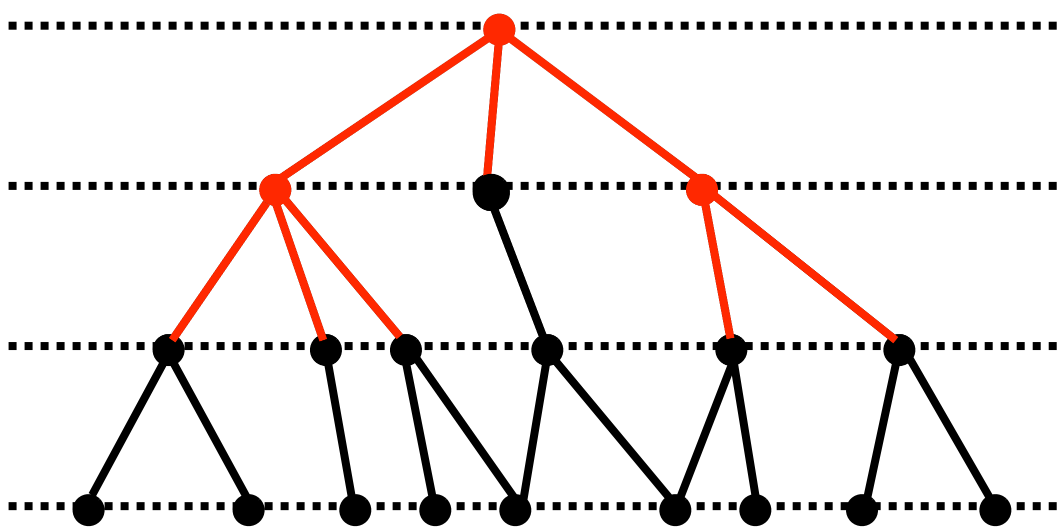
$W_0 \models^? \mathrm{AG}\varphi \Rrightarrow$

$W_1 \models^? \mathrm{AG}\varphi$

W_2

W_3

Runtime Verification



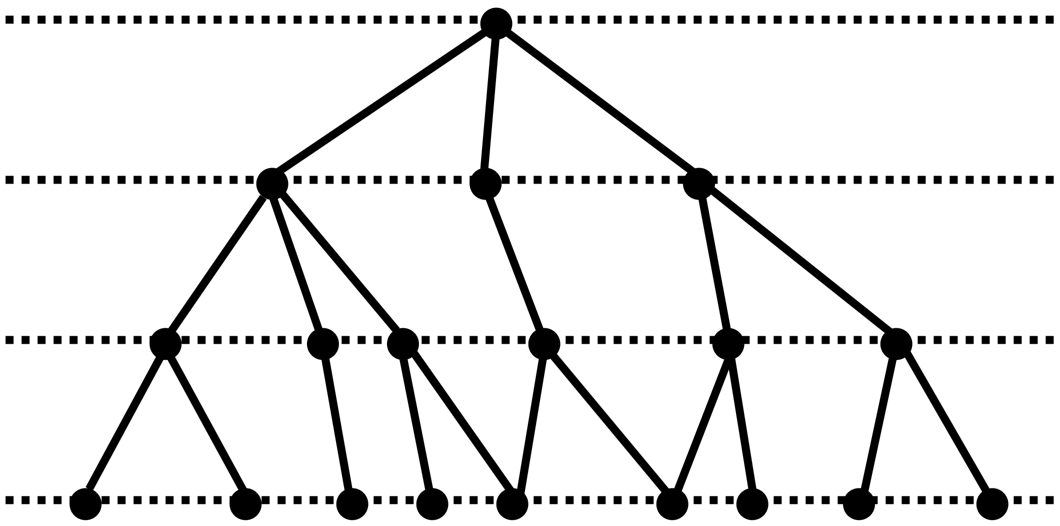
$W_0 \models^{?} AG\varphi \Rrightarrow$

$W_1 \models^{?} AG\varphi \Rrightarrow$

$W_2 \not\models AG\varphi$

W_3

Runtime Verification



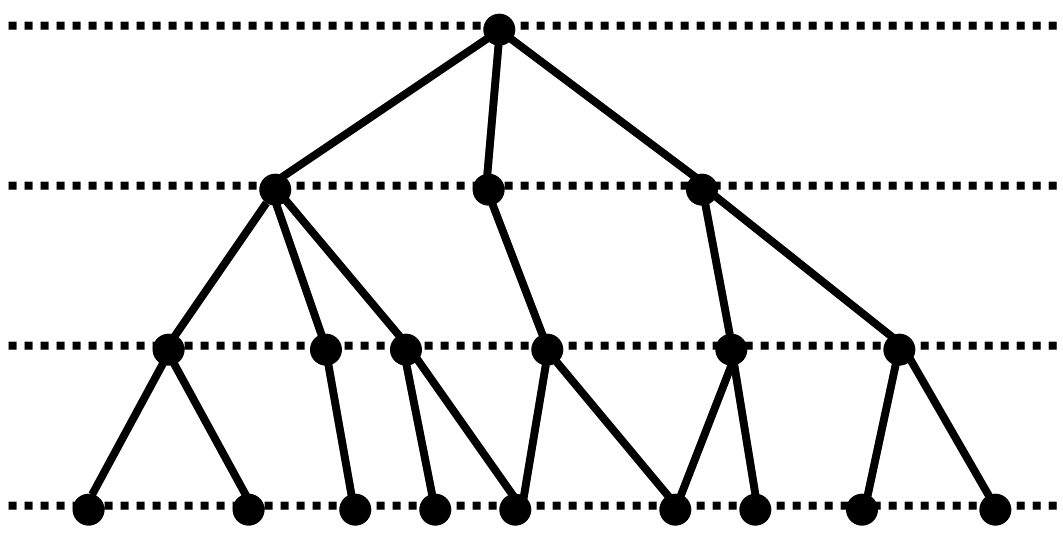
$W_0 \models^? \text{EF}\varphi$

W_1

W_2

W_3

Runtime Verification



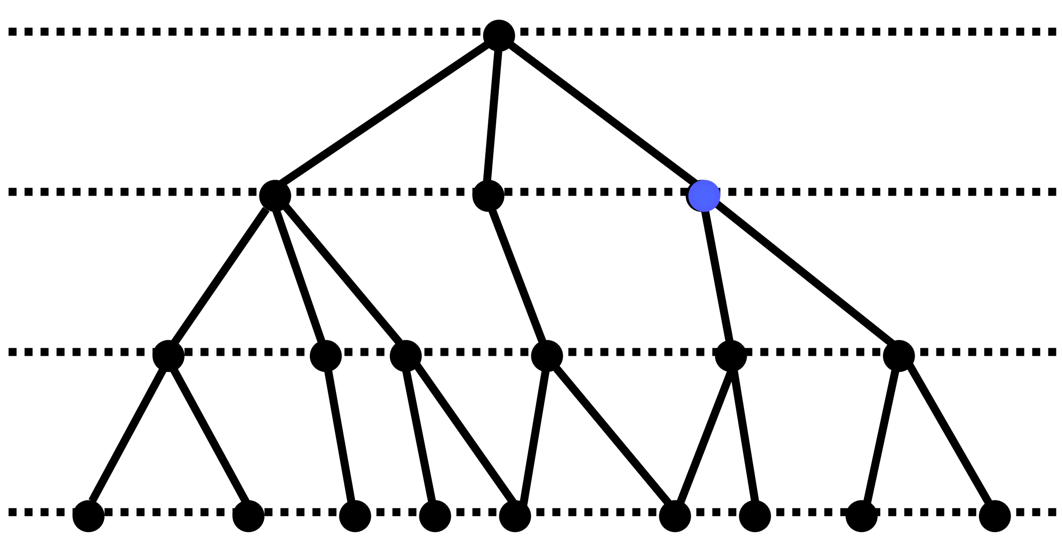
$W_0 \models^? \mathrm{EF}\varphi \Rrightarrow$

$W_1 \models^? \mathrm{EF}\varphi$

W_2

W_3

Runtime Verification



$W_0 \models^{?} \mathrm{EF}\varphi \Rrightarrow$

$W_1 \models^{?} \mathrm{EF}\varphi \Rrightarrow$

$W_2 \models \mathrm{EF}\varphi$

W_3

- An Abstract Biochemical Calculus
- Port Graph Rewriting
- A Biochemical Calculus Based on Strategic Port Graph Rewriting
- Runtime Verification in the Biochemical Calculus
- Conclusions and Perspectives

Conclusions

Aim: develop better models of living phenomena and new biologically-inspired computational models

Conclusions

Aim: develop better models of living phenomena and new biologically-inspired computational models

- a higher-order biochemical calculus
- port graphs
- simulation and verification
- applications to protein-protein interactions and autonomous systems

Conclusions

Aim: develop better models of living phenomena and new biologically-inspired computational models

- a higher-order biochemical calculus
- port graphs
- simulation and verification
- applications to protein-protein interactions and autonomous systems

Strategies!

Perspectives

- express other or new strategies, instantiate with various structures (bigraphs)
- develop efficient simulation methods with graphical interface
- other types of bio-molecular interactions, stochastic aspects
- applicability in modeling synthetic biology

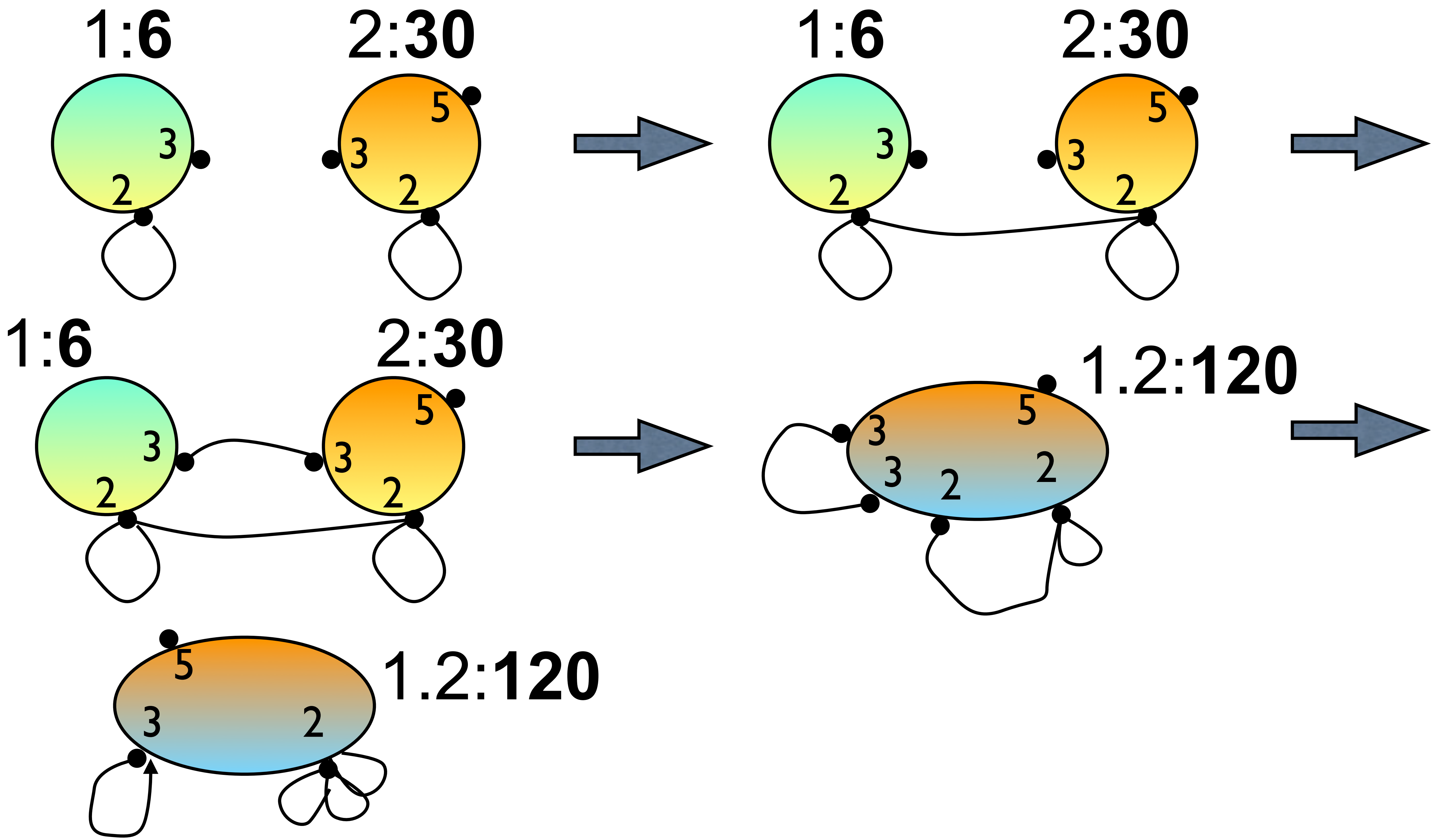
Merci de votre attention!

Publications

With Hélène Kirchner:

- ★ A Higher-Order Graph Calculus for Autonomic Computing - *GTCIT'08*
- ★ A Biochemical Calculus Based on Strategic Graph Rewriting - *AB'08*
- ★ Strategic Port Graph Rewriting for Autonomic Computing - *TFIT'08*
- ★ Graph Rewriting and Strategies for Modeling Biochemical Networks - *NCA'07*
- ★ A Rewriting Calculus for Multigraphs with Ports - *RULE'07*

- Strategy-Based Proof Calculus for Membrane Systems - *WRLA'08 (with Dorel Lucanu)*
- A rewriting logic framework for operational semantics of membrane systems - *TCS'07 (with Gabriel Ciobanu and Dorel Lucanu)*
- Expressing Control Mechanisms of Membranes by Rewriting Strategies - *WMC'06 (with Gabriel Ciobanu and Dorel Lucanu)*



repeat(CreateBondBetweenIdenticalPorts);
FusionNodes;
repeat(FusionIdenticalPorts)