

# MPRI 2.19 Biochemical Programming

Rule-based Modeling

**Static analysis**

**Jérôme Feret**

DIENS (ÉNS, CNRS, INRIA, PSL)



[kappalanguage.org](http://kappalanguage.org)

Monday, the 16th of December, 2024

# In this talk...

We illustrate the following concepts:

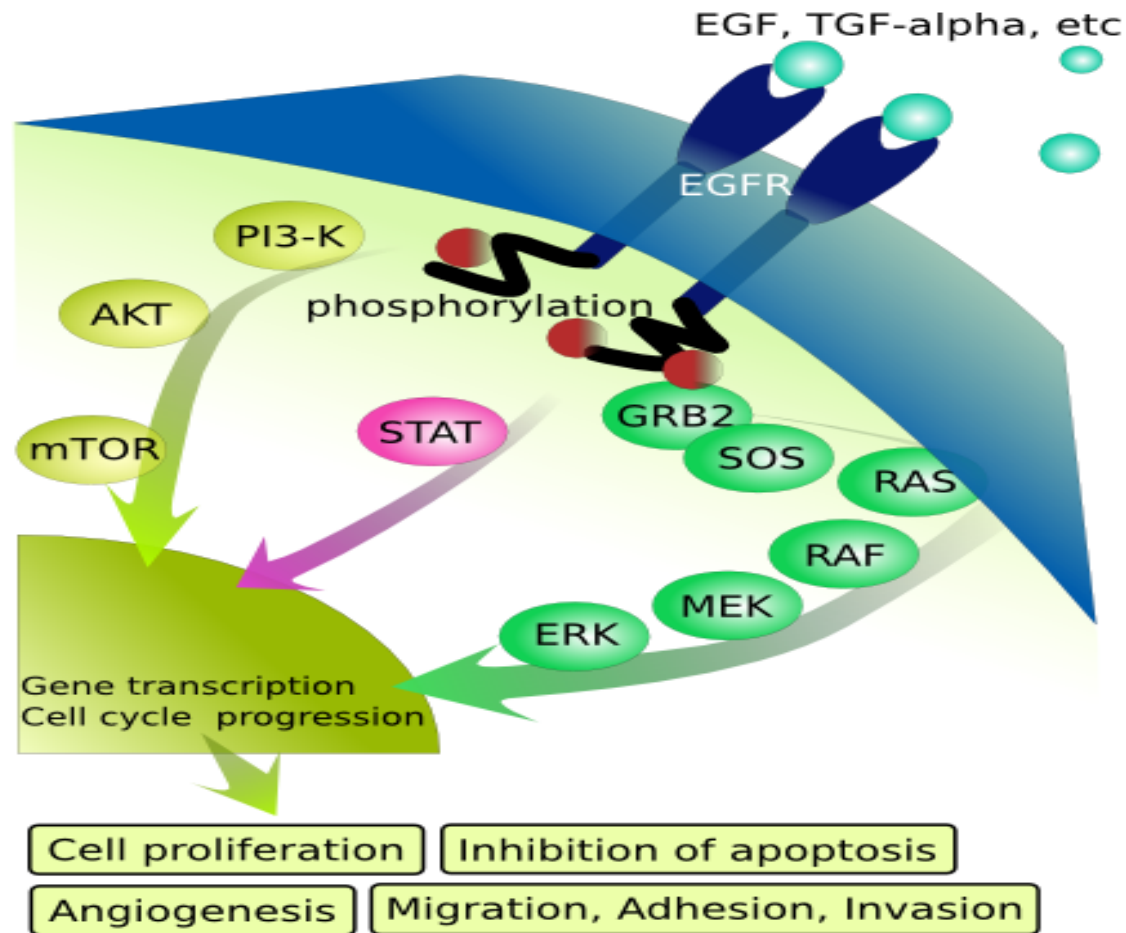
- Galois connections:
  - the upper closure operator  $\gamma \circ \alpha$ ,
  - the lower closure operator  $\alpha \circ \gamma$ ;
- soundness:
  - the abstraction forgets no behavior;
- completeness:
  - sufficient conditions that ensure the absence of false positive;

on an abstraction of the reachable connected components in a site-graph rewriting language.

# Overview

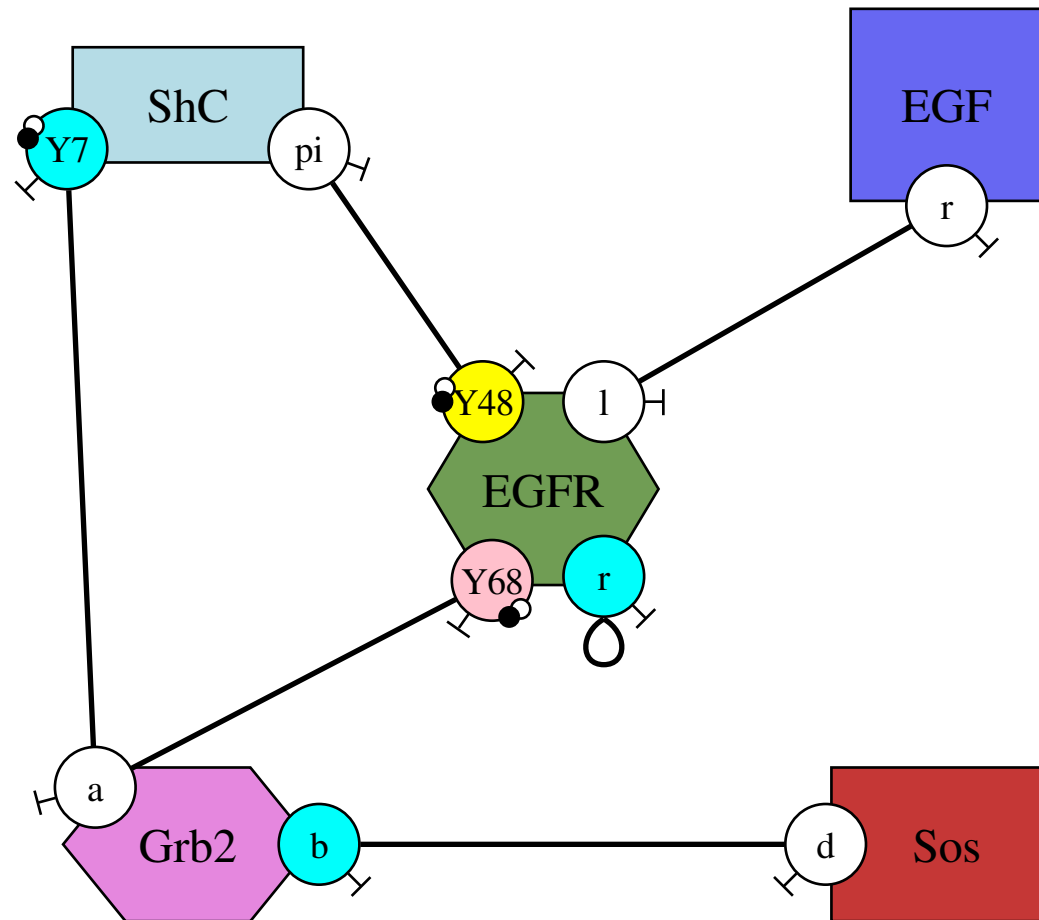
1. Introduction
2. Abstraction: Local views
3. Completeness: false positives?
4. Local fragment of Kappa
5. Other abstract domains
6. Decontextualization
7. Conclusion

# Signaling Pathways

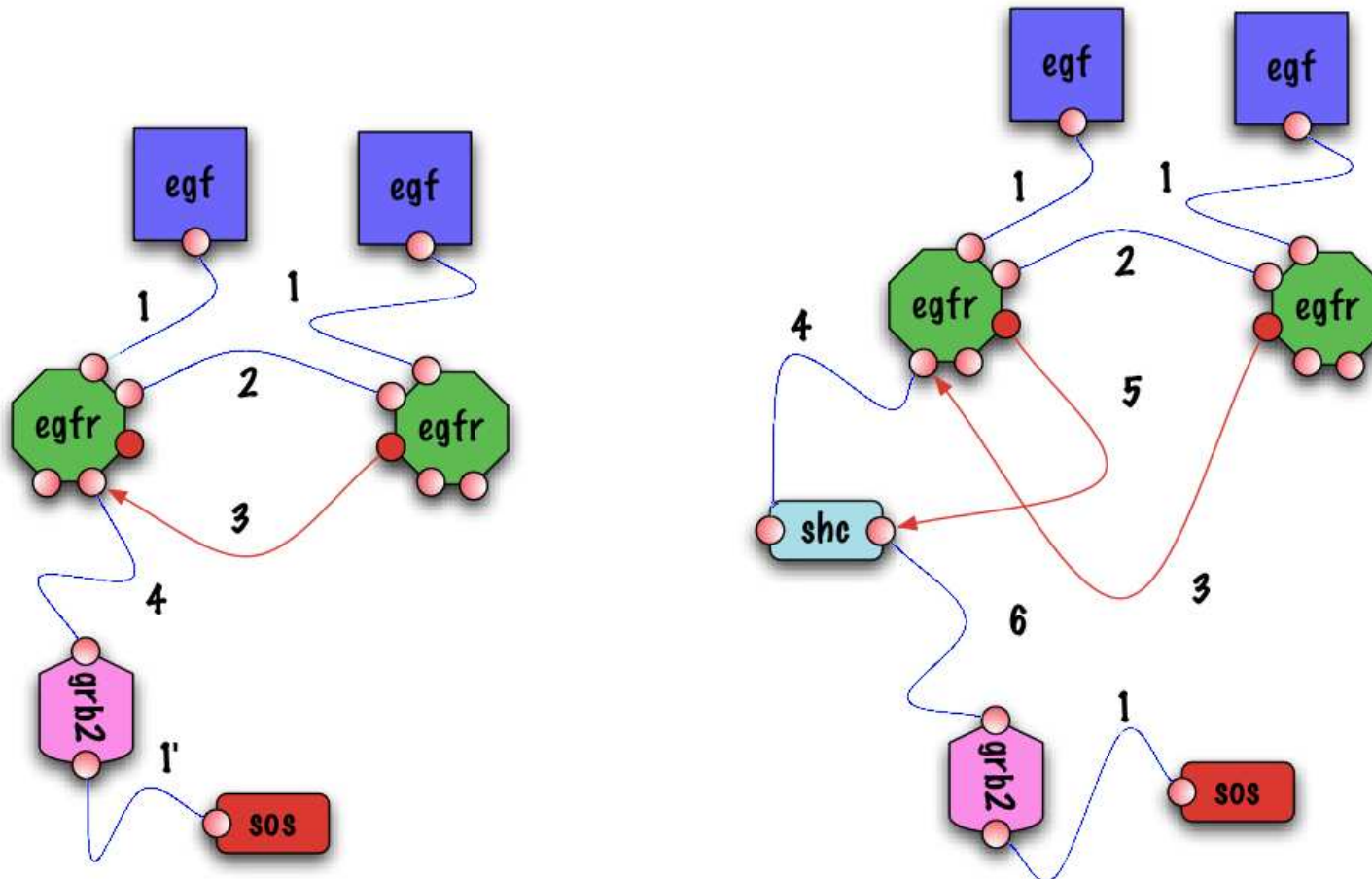


Eikuch, 2007

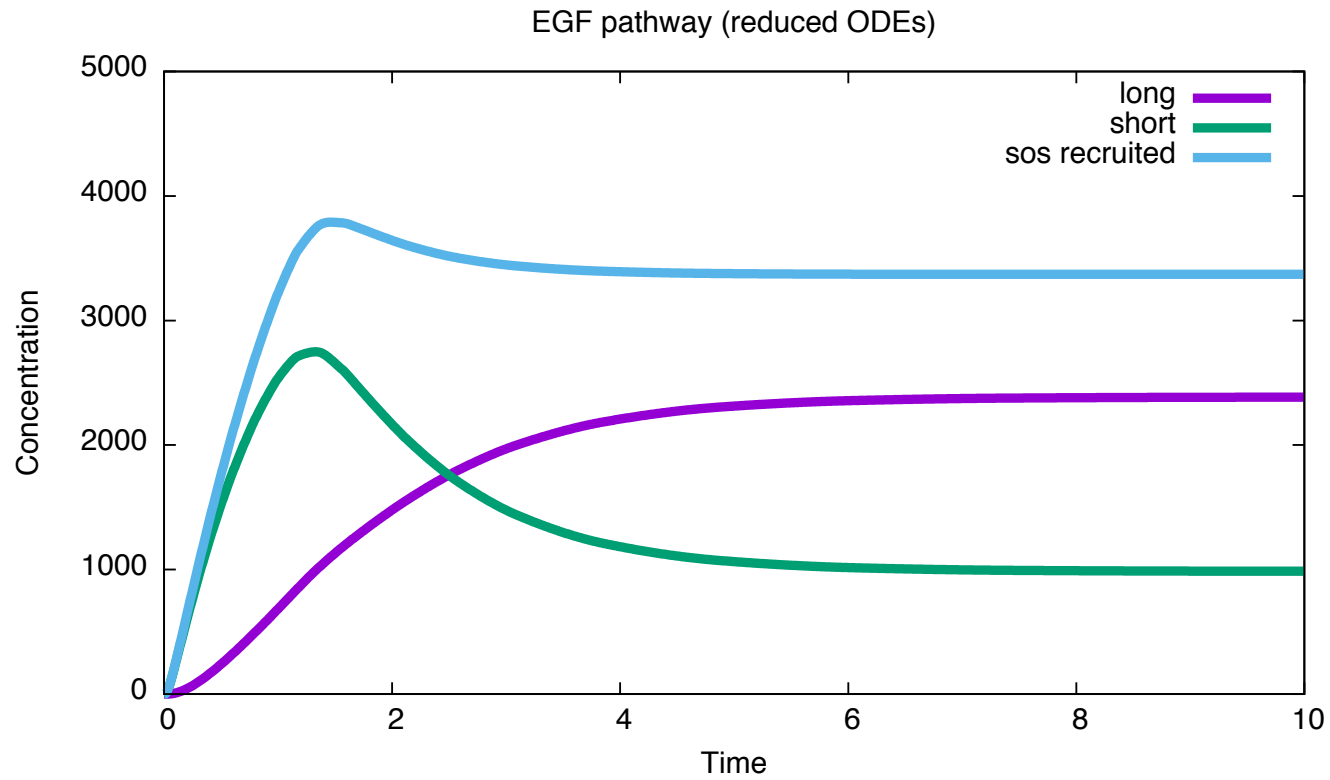
# Contact map



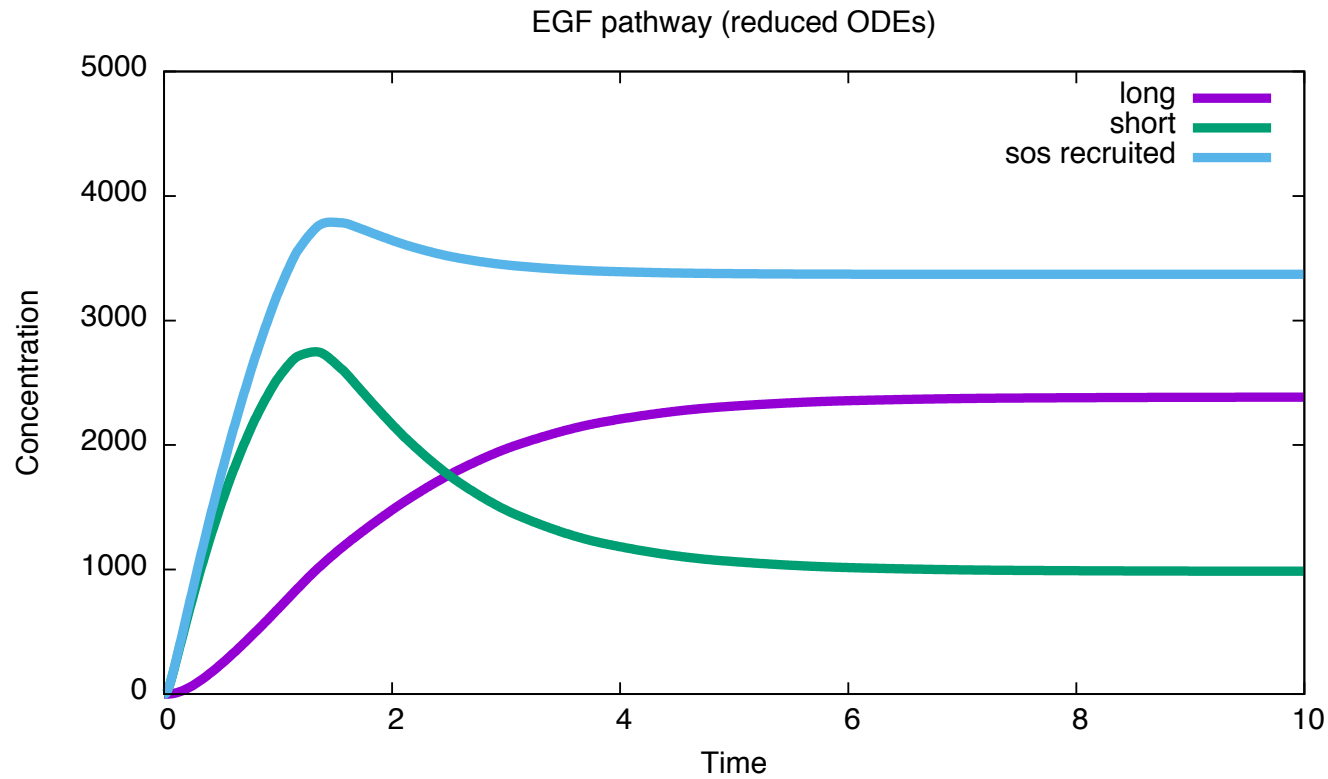
# Causal traces



# ODE semantics



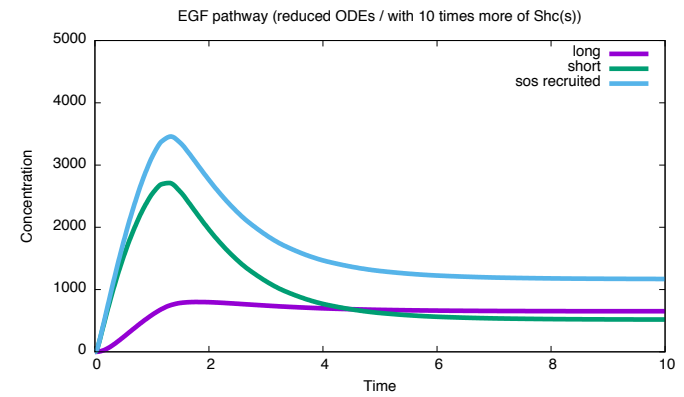
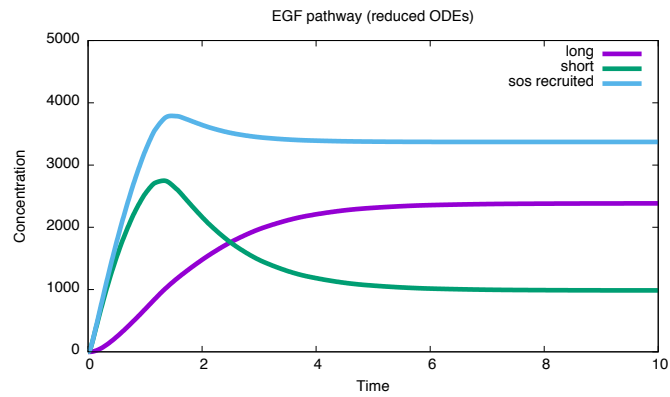
# ODE semantics



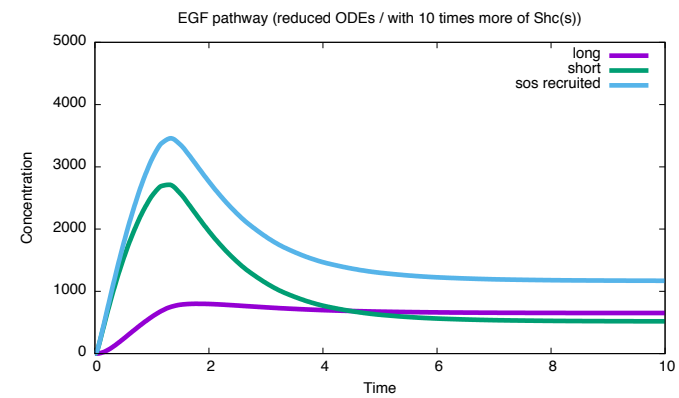
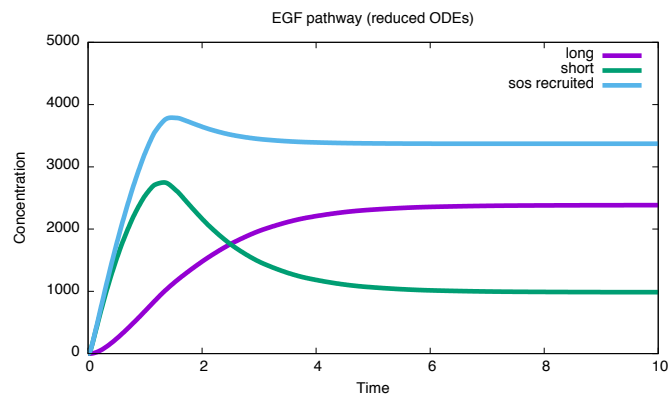
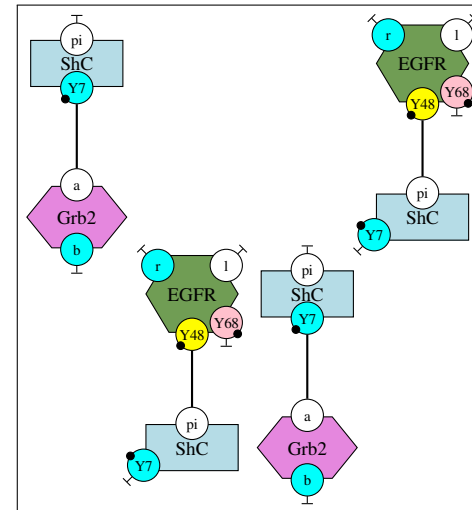
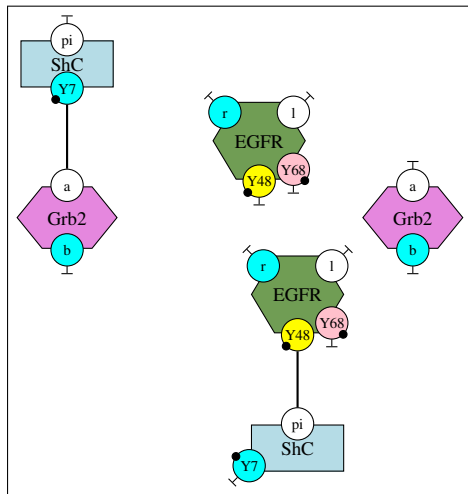
What will happen if more Shc(s) is put in the system?



# ODE semantics



# Crowding effect



# Early EGF example

## egf rules 1

protein shorthands: E:=egf, R:=egfr, So:=Sos, Sh:=Sh, G:=grb2  
site abbreviations & fusions: Y68:=Y1068, Y48:=Y1148/73, Y7:=Y317,  $\pi$ :=PTB/SH2

- Ligand-receptor binding, receptor dimerisation, rtk x-phosph, & de-phosph

- 01:  $R(l,r), E(r) \leftrightarrow R(l^1,r), E(r^1)$
- 02:  $R(l^1,r), R(l^2,r) \leftrightarrow R(l^1,r^3), R(l^2,r^3)$
- 03:  $R(r^1,Y68) \rightarrow R(r^1,Y68^p)$   
 $R(Y68^p) \rightarrow R(Y68)$
- 04:  $R(r^1,Y48) \rightarrow R(r^1,Y48^p)$   
 $R(Y48^p) \rightarrow R(Y48)$

receptor type:  $R(l,r,Y68,Y48)$

- Sh x-phosph & de-phosph

- 14:  $R(r^2,Y48^{p1}), Sh(\pi^1,Y7) \rightarrow R(r^2,Y48^{p1}), Sh(\pi^1,Y7^p)$
- ??:  $Sh(\pi^1,Y7^p) \rightarrow Sh(\pi^1,Y7)$
- 16:  $Sh(\pi,Y7^p) \rightarrow Sh(\pi,Y7)$

refined from  
 $Sh(Y7^p) \rightarrow Sh(Y7)$

- Y68-G binding

- 09:  $R(Y68^p), G(a,b) \leftrightarrow R(Y68^{p1})+G(a^1,b)$
- 11:  $R(Y68^p), G(a,b^2) \leftrightarrow R(Y68^{p1})+G(a^1,b^2)$

refined from  
 $R(Y68^p)+G(a) \leftrightarrow R(Y68^{p1})+G(a^1)$

# Early EGF example

## egf rules 2

refined from  
 $So(d) + G(b) \leftrightarrow So(d^1) + G(b^1)$

interface note: highlight  
the interacting parts

- **G-So binding**

- 10:  $R(Y68^{p1}), G(a^1, b), So(d) \leftrightarrow R(Y68^{p1}), G(a^1, b^2), So(d^2)$
- 12:  $G(a, b), So(d) \leftrightarrow G(a, b^1), So(d^1)$
- 22:  $Sh(\pi, Y7^{p2}), G(a^2, b), So(d) \leftrightarrow Sh(\pi, Y7^{p2}), G(a^2, b^1), S(d^1)$
- 19:  $Sh(\pi^1, Y7^{p2}), G(a^2, b), So(d) \leftrightarrow Sh(\pi^1, Y7^{p2}), G(a^2, b^1), S(d^1)$

- **Y48-Sh binding**

- 13:  $R(Y48^p), Sh(\pi, Y7) \leftrightarrow R(Y48^{p1}), Sh(\pi^1, Y7)$
- 15:  $R(Y48^p), Sh(\pi, Y7^p) \leftrightarrow R(Y48^{p1}), Sh(\pi^1, Y7^p)$
- 18:  $R(Y48^p), Sh(\pi, Y7^{p1}), G(a^1, b) \leftrightarrow R(Y48^{p2}), Sh(\pi^2, Y7^{p1}), G(a^1, b)$
- 20:  $R(Y48^p), Sh(\pi, Y7^{p1}), G(a^1, b^3), S(d^3) \leftrightarrow R(Y48^{p2}), Sh(\pi^2, Y7^{p1}), G(a^1, b^3), S(d^3)$

refined from  
 $R(Y48^p) + Sh(\pi) \leftrightarrow R(Y48^{p1}) + Sh(\pi^1)$

why not simply  $G(b^3)??$

- **Sh-G binding**

- 17:  $R(Y48^{p1}), Sh(\pi^1, Y7^p), G(a, b) \leftrightarrow R(Y48^{p1}), Sh(\pi^1, Y7^{p2}), G(a^2, b)$
- 21:  $Sh(\pi, Y7^p), G(a, b) \leftrightarrow Sh(\pi, Y7^{p1}), G(a^1, b)$
- 23:  $Sh(\pi, Y7^p), G(a, b^2) \leftrightarrow Sh(\pi, Y7^{p1}), G(a^1, b^2)$
- 24:  $R(Y48^{p1}), Sh(\pi^1, Y7^p), G(a, b^3), S(d^3) \leftrightarrow R(Y48^{p1}), Sh(\pi^1, Y7^{p2}), G(a^2, b^3), S(d^3)$

refined from  
 $Sh(\pi), G(a) \leftrightarrow Sh(\pi^1), G(a^1)$



# Properties of interest

## 1. Show the absence of modeling errors:

- detect dead rules;
- detect overlapping rules;
- detect non exhaustive interactions;
- detect rules with ambiguous molecularity.

## 2. Get idiomatic description of the networks:

- capture causality;
- capture potential interactions;
- capture relationships between site states;
- simplify rules.

## 3. Allow fast simulation:

- capture accurate approximation of the wake-up relation.

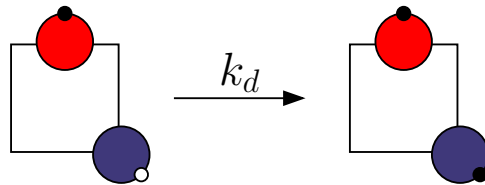
# Overview

1. Introduction
2. **Abstraction: Local views**
3. Completeness: false positives?
4. Local fragment of Kappa
5. Other abstract domains
6. Decontextualization
7. Conclusion

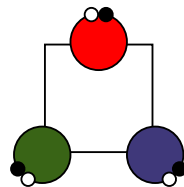
# Concrete semantics

A rule is a symbolic representation of a multi-set of reactions.

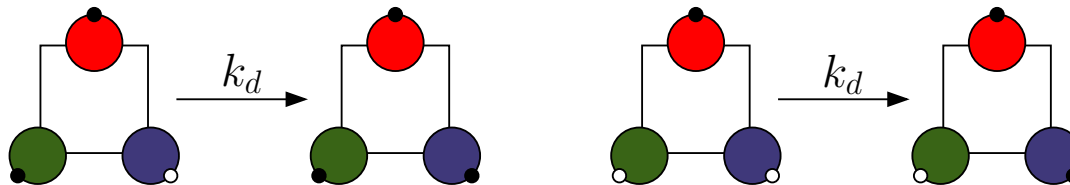
For instance, the rule:



within a model with the following signature:



denotes the following two reactions:



# Set of reachable chemical species

Let  $\mathcal{R} = \{R_i\}$  be a set of rules.

Let  $Species$  be the set of all chemical species ( $C, c_1, c'_1, \dots, c_k, c'_k, \dots \in Species$ ).

Let  $Species_0$  be the set of initial chemical species.

We are interested in  $Species_\omega$  the set of all chemical species that can be constructed in one or several applications of the reactions induced by the rules in  $\mathcal{R}$ , starting from the set  $Species_0$  of initial chemical species.

(We do not care about the number of occurrences of each chemical species).



# Inductive definition

We define the mapping  $\mathbb{F}$  as follows:

$$\mathbb{F} : \begin{cases} \wp(\textit{Species}) & \rightarrow \wp(\textit{Species}) \\ X & \mapsto X \cup \left\{ c'_j \mid \begin{array}{l} \exists R_k \in \mathcal{R}, c_1, \dots, c_m \in X, \\ c_1, \dots, c_m \rightarrow_{R_k} c'_1, \dots, c'_n \end{array} \right\} \end{cases}.$$

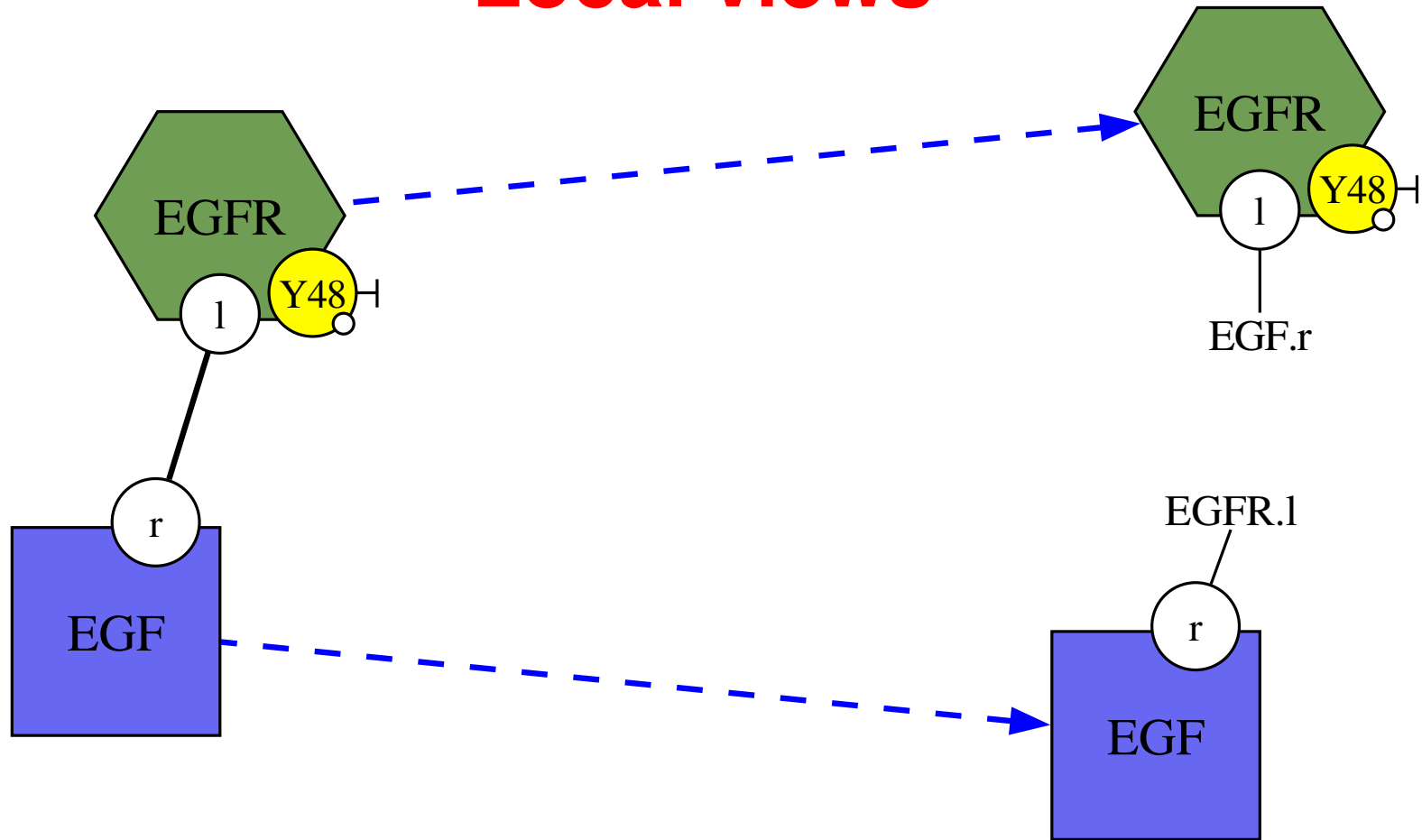
The set  $\wp(\textit{Species})$  is a complete lattice.

The mapping  $\mathbb{F}$  is an extensive  $\cup$ -complete morphism.

We define the set of reachable chemical species as follows:

$$\textit{Species}_\omega = \bigcup \{ \mathbb{F}^n(\textit{Species}_0) \mid n \in \mathbb{N} \}.$$

# Local views



$$\alpha(\{\text{EGFR}(\text{Y48}\{u\}[\cdot] \mid [1]) \text{ EGF}(r[1])\}) = \{\text{EGFR}(\text{Y48}\{u\}[\cdot] \mid [r.\text{EGF}]); \text{EGF}(r[\mid.\text{EGFR}])\}.$$

# Galois connection

Let  $Local\_view$  be the set of all local views.

Let  $\alpha \in \wp(Species) \rightarrow \wp(Local\_view)$  be the function that maps any set of chemical species into the set of their local views.

The set  $\wp(Local\_view)$  is a complete lattice.  
The function  $\alpha$  is a  $\cup$ -complete morphism.

Thus, it defines a Galois connection:

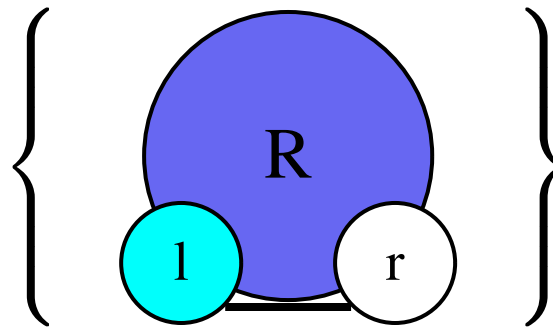
$$\wp(Species) \xrightleftharpoons[\alpha]{\gamma} \wp(Local\_view).$$

(The function  $\gamma$  maps a set of local views into the set of complexes that can be built with these local views).

$$\gamma \circ \alpha$$

$\gamma \circ \alpha$  is an upper closure operator: it abstracts away some information.

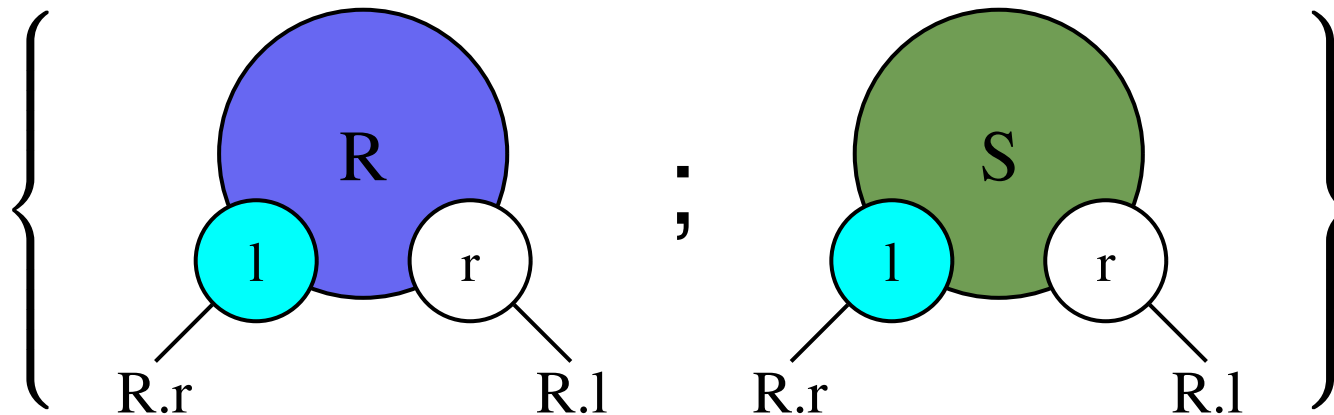
Guess the image of the following set of chemical species ?



$$\alpha \circ \gamma$$

$\alpha \circ \gamma$  is a lower closure operator: it simplifies (or reduces) constraints.

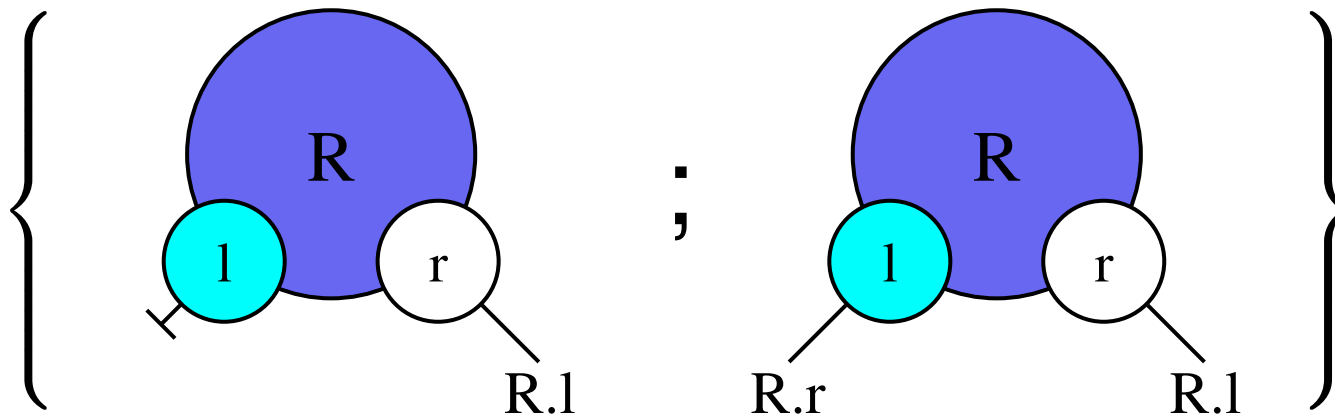
Guess the image of the following set of local views ?



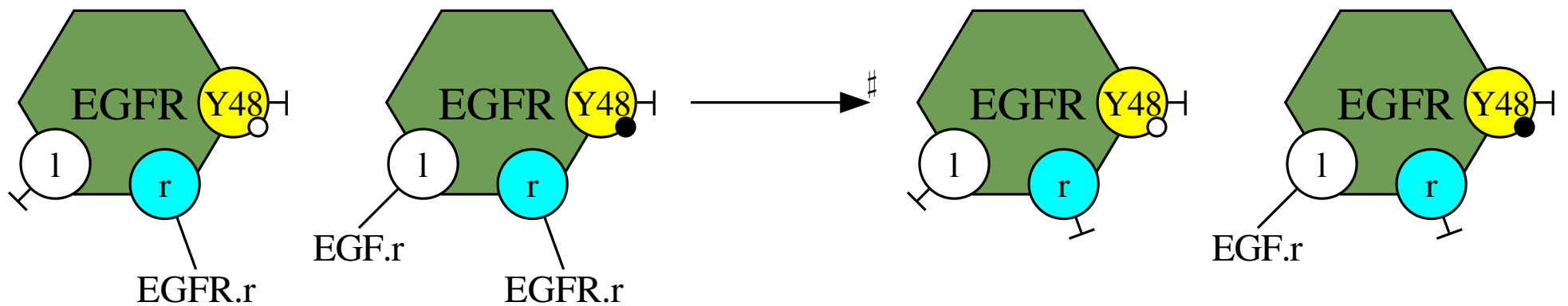
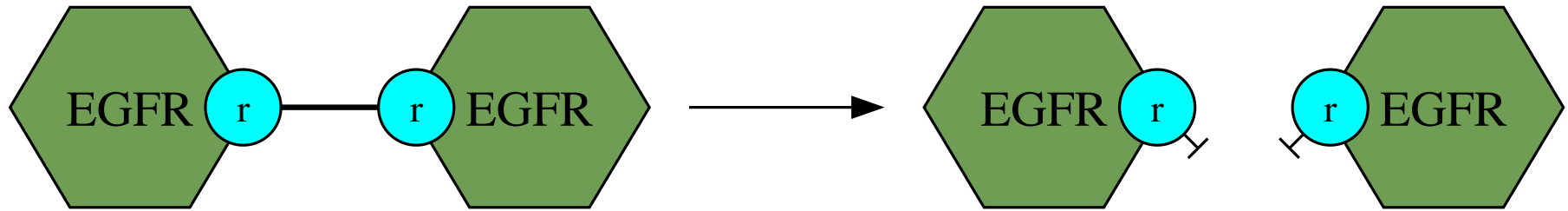
# One more question

$\alpha \circ \gamma$  is a lower closure operator: it simplifies (or reduces) constraints.

Guess the image of the following set of local views ?



# Abstract reactions



# Abstract counterpart to $\mathbb{F}$

We define  $\mathbb{F}^\#$  as:

$$\mathbb{F}^\# : \begin{cases} \wp(\text{Local\_view}) & \rightarrow \wp(\text{Local\_view}) \\ Y & \mapsto Y \cup \left\{ V_j \mid \begin{array}{l} \exists R_k \in \mathcal{R}, l_{v_1}, \dots, l_{v_m} \in Y, \\ l_{v_1}, \dots, l_{v_m} \rightarrow_{R_k}^\# l'_{v_1}, \dots, l'_{v_n} \end{array} \right\}. \end{cases}$$

We have:

- $\mathbb{F}^\#$  is extensive;
- $\mathbb{F}^\#$  is monotonic;
- $\mathbb{F} \circ \gamma \subseteq \gamma \circ \mathbb{F}^\#$ ;
- $\mathbb{F}^\# \circ \alpha = \alpha \circ \mathbb{F} \circ \gamma \circ \alpha$  (we will see later why).



# Soundness

**Theorem 1** Let:

1.  $(D, \subseteq, \cup)$  and  $(D^\#, \sqsubseteq, \sqcup)$  be chain-complete partial orders;
2.  $D \xrightleftharpoons[\alpha]{\gamma} D^\#$  be a Galois connection;
3.  $\mathbb{F} \in D \rightarrow D$  and  $\mathbb{F}^\# \in D^\# \rightarrow D^\#$  be monotonic mappings such that:  
 $\mathbb{F} \circ \gamma \subseteq \gamma \circ \mathbb{F}^\#$ ;
4.  $X_0 \in D$  be an element such that:  $X_0 \subseteq \mathbb{F}(X_0)$ ;

Then:

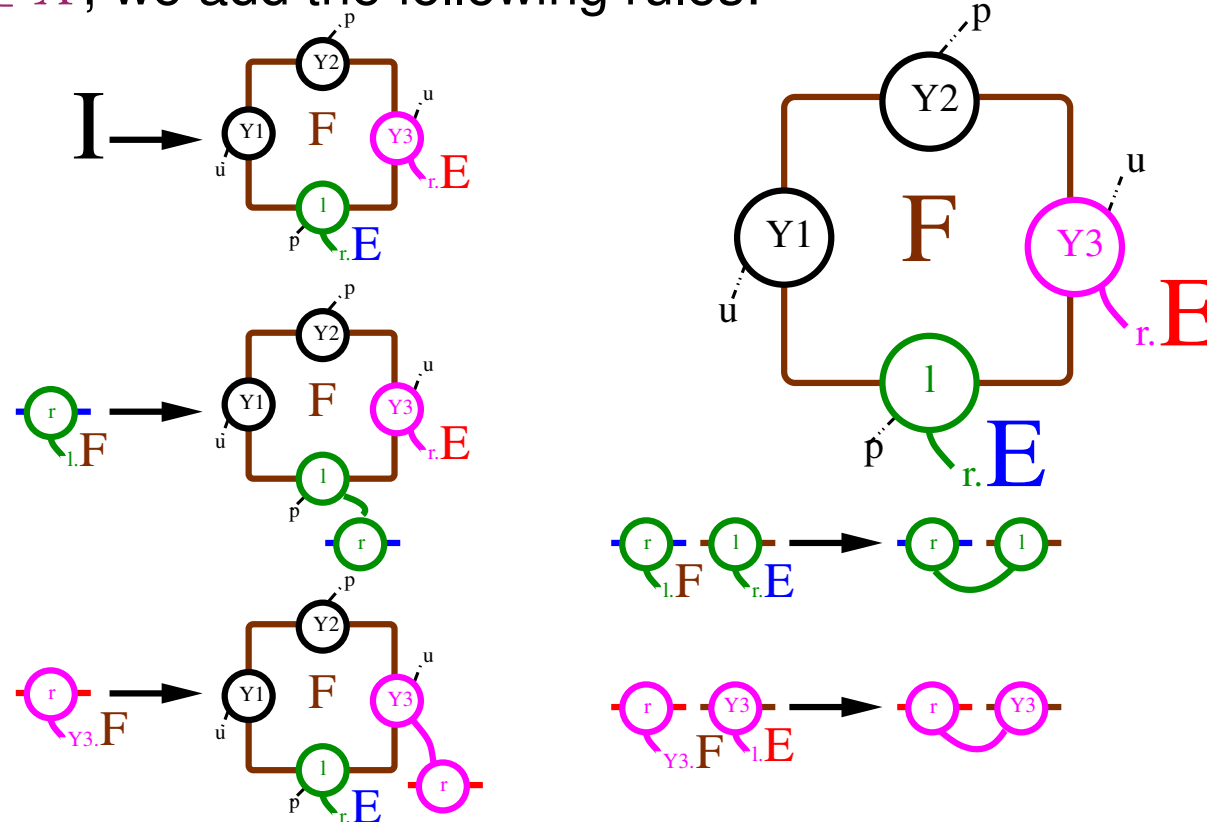
1. both  $\text{lfp}_{X_0} \mathbb{F}$  and  $\text{lfp}_{\alpha(X_0)} \mathbb{F}^\#$  exist,
2.  $\text{lfp}_{X_0} \mathbb{F} \subseteq \gamma(\text{lfp}_{\alpha(X_0)} \mathbb{F}^\#)$ .

# Overview

1. Introduction
2. Abstraction: Local views
3. Completeness: false positives?
4. Local fragment of Kappa
5. Other abstract domains
6. Decontextualization
7. Conclusion

# From views to species

For any  $X \in \wp(\text{Local\_view})$ ,  $\gamma(X)$  is given by a rewrite system:  
 For any  $lv \in X$ , we add the following rules:



$I$  and semi-links are non-terminal.  
 $I$  is the initial symbol.

# Pumping lemma

- We use this rewrite system to enumerate the chemical species of  $\gamma(X)$ .
- There are two cases:
  1. either there is a finite number of rewrite sequences;
  2. or we encounter cyclic derivations  
i.e. an open chemical species with a cycle of the following form:

$R.l-r.E \dots R.l-r.E$

can be built.

- We only enumerate chemical species that are reached through an acyclic rewriting computation.
- It turns out that: if  $X \in \alpha(\wp(\textit{Species}))$  then each rewrite sequence is the prefix of a terminating rewrite sequence.  
(So there is an unbounded number of species if, and only if, there is an unbounded number of rewrite sequences.)

# Examples

1. Make the demo for egf
2. Make the demo for fgf
3. Make the demo for Global invariants

# Counting chemical species

Given a set of local views  $X$ , we can easily count the number of species in  $\gamma(X)$  by using the following lemmas:

**Lemma 1 (rigidity)** An embedding between two connected components is fully characterized by the image of one agent.

**Lemma 2 (automorphism)** If  $\gamma(X)$  is finite, then for any  $C \in \gamma(X)$ :

- $C$  has at most two automorphisms;
- if  $C$  has two automorphisms, then  $C$  has a bond of the form  $R.r - r.R$ . Moreover one automorphism swaps the two  $R$  of this bond.

**Lemma 3 (Euler)** If a chemical species has no cycle, then it has an agent with only one site.

## sketch the algorithm

# Which information is abstracted away ?

Our analysis is exact (no false positive):

- for EGF cascade (356 chemical species);
- for FGF cascade (79080 chemical species);
- for SBF cascade (around  $10^{19}$  chemical species).

We know how to build systems with false positives. . .

. . .but they seem to be biologically meaningless.

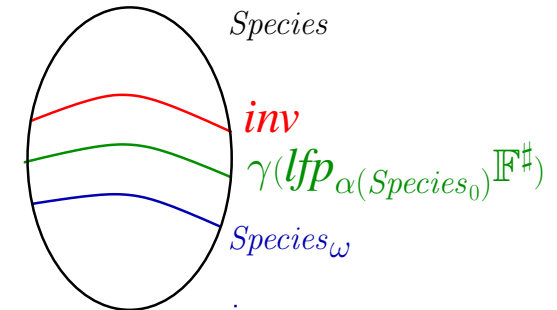
This raises the following issues:

- Can we characterize which information is abstracted away ?
- Which is the form of the systems, for which we have no false positive ?
- Do we learn something about the biological systems that we describe ?

# Which information is abstracted away ?

**Theorem 2** We suppose that:

1.  $(D, \subseteq)$  be a partial order;
2.  $(D^\sharp, \sqsubseteq, \sqcup)$  be chain-complete partial order;
3.  $D \xrightleftharpoons[\alpha]{\gamma} D^\sharp$  be a Galois connection;
4.  $\mathbb{F} \in D \rightarrow D$  and  $\mathbb{F}^\sharp \in D^\sharp \rightarrow D^\sharp$  are monotonic;
5.  $\mathbb{F} \circ \gamma \subseteq \gamma \circ \mathbb{F}^\sharp$ ;
6.  $X_0, inv \in D$  such that:
  - $X_0 \subseteq \mathbb{F}(X_0) \subseteq \mathbb{F}(inv) \subseteq inv$ ,
  - $inv = \gamma(\alpha(inv))$ ,
  - and  $\alpha(\mathbb{F}(inv)) = \mathbb{F}^\sharp(\alpha(inv))$ ;



Then,  $lfp_{\alpha(X_0)}\mathbb{F}^\sharp$  exists and  $\gamma(lfp_{\alpha(X_0)}\mathbb{F}^\sharp) \subseteq inv$ .



# Proof I/III

We have already seen (previous lectures) that:

1.  $\text{lfp}_{\alpha(X_0)} \mathbb{F}^\#$  exists;
2. there exists an ordinal  $\delta$  such that  $\text{lfp}_{\alpha(X_0)} \mathbb{F}^\# = \mathbb{F}^{\#\delta}(\alpha(X_0))$ .

## Proof II/III

Let us show that  $\gamma(\text{Ifp}_{\alpha(X_0)}\mathbb{F}^\#) \subseteq \text{inv}$ .

Let us prove instead by induction over  $\delta$  that  $\mathbb{F}^{\#\delta}(\alpha(X_0)) \subseteq \alpha(\text{inv})$ .

- If  $Y \in D^\#$  is an element such that  $Y \subseteq \alpha(\text{inv})$ ,  
 $\mathbb{F}^\#(Y) \subseteq \mathbb{F}^\#(\alpha(\text{inv}))$  ( $\mathbb{F}^\#$  is mon)  
 $\mathbb{F}^\#(\alpha(\text{inv})) = \alpha(\mathbb{F}(\text{inv}))$  (assumption)  
 $\alpha(\mathbb{F}(\text{inv})) \subseteq \alpha(\text{inv})$ . ( $\alpha$  is mon and  $\text{inv}$  is a post)

Thus:  $\mathbb{F}^\#(Y) \subseteq \alpha(\text{inv})$

- If  $Y_i \in D^{\#I}$  is a chain of elements such that  $Y_i \subseteq \alpha(\text{inv})$  for any  $i \in I$ ,  
then,  $\sqcup Y_i \subseteq \alpha(\text{inv})$  (lub).

So:  $\mathbb{F}^{\#\delta}(\alpha(X_0)) \subseteq \alpha(\text{inv})$ .

# Proof III/III

We have:

$$\mathbb{F}^{\# \delta}(\alpha(X_0)) \sqsubseteq \alpha(\textit{inv}).$$

Since  $\gamma$  is monotonic:

$$\gamma(\mathbb{F}^{\# \delta}(\alpha(X_0))) \subseteq \gamma(\alpha(\textit{inv})).$$

But, by assumption,  $\gamma(\alpha(\textit{inv})) = \textit{inv}$ .

Thus,

$$\gamma(\mathbb{F}^{\# \delta}(\alpha(X_0))) \subseteq \textit{inv}.$$

# When is there no false positive ?

**Theorem 3** We suppose that:

1.  $(D, \subseteq, \cup)$  and  $(D^\#, \sqsubseteq, \sqcup)$  are chain-complete partial orders;
2.  $(D, \subseteq) \xrightleftharpoons[\alpha]{\gamma} (D^\#, \sqsubseteq)$  is a Galois connection;
3.  $\mathbb{F} : D \rightarrow D$  is a monotonic map;
4.  $X_0$  is a concrete element such that  $X_0 \subseteq \mathbb{F}(X_0)$ ;
5.  $\mathbb{F} \circ \gamma \subseteq \gamma \circ \mathbb{F}^\#$ ;
6.  $\mathbb{F}^\# \circ \alpha = \alpha \circ \mathbb{F} \circ \gamma \circ \alpha$ .

Then:

- $\text{lfp}_{X_0} \mathbb{F}$  and  $\text{lfp}_{\alpha(X_0)} \mathbb{F}^\#$  exist;
- $\text{lfp}_{X_0} \mathbb{F} = \gamma(\alpha(\text{lfp}_{X_0} \mathbb{F})) \iff \text{lfp}_{X_0} \mathbb{F} = \gamma(\text{lfp}_{\alpha(X_0)} \mathbb{F}^\#)$ .

We need to understand under which assumptions  $\text{lfp}_{X_0} \mathbb{F} = \gamma(\alpha(\text{lfp}_{X_0} \mathbb{F}))$ .

# Local set of chemical species

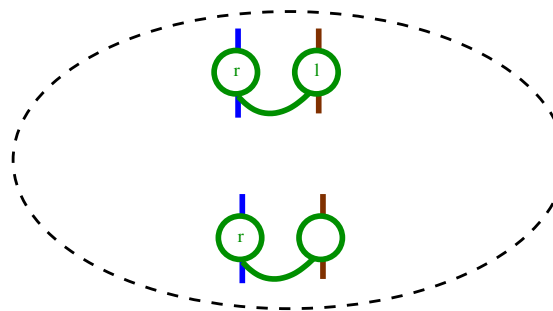
**Definition 1** We say that a set  $X \in \wp(\textit{Species})$  of chemical species is local if and only if  $X \in \gamma(\wp(\textit{Local\_view}))$ .

(ie. a set  $X$  is local if and only if  $X$  is exactly the set of all the species that are generated by a given set of local views.)

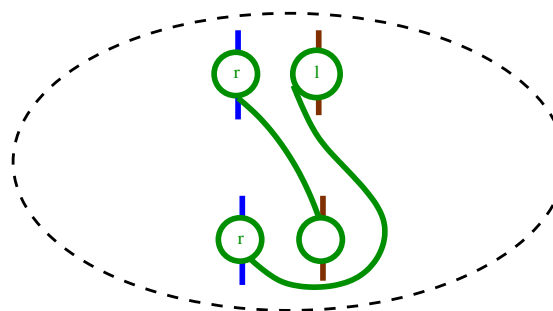
# Swapping relation

We define the binary relation  $\overset{\text{SWAP}}{\sim}$  among tuples  $\textit{Species}^*$  of chemical species.  
 We say that  $(C_1, \dots, C_m) \overset{\text{SWAP}}{\sim} (D_1, \dots, D_n)$  if and only if:

$(C_1, \dots, C_m)$  matches with



while  $(D_1, \dots, D_n)$  matches with



# Swapping closure

**Theorem 4** Let  $X \in \wp(\textit{Species})$  be a set of chemical species.

The two following assertions are equivalent:

1.  $X = \gamma(\alpha(X))$ ;
2. for any tuples  $(C_i), (D_j) \in \textit{Species}^*$  such that:
  - $(C_i) \in X^*$ ,
  - and  $(C_i) \stackrel{\text{SWAP}}{\sim} (D_j)$ ;we have  $(D_j) \in X^*$ .

# Proof (easier implication way)

If:

- $X = \gamma(\alpha(X))$ ,
- $(C_i)_{i \in I} \in X^*$ ,
- and  $(C_i)_{i \in I} \overset{\text{SWAP}}{\sim} (D_j)_{j \in J}$ ;

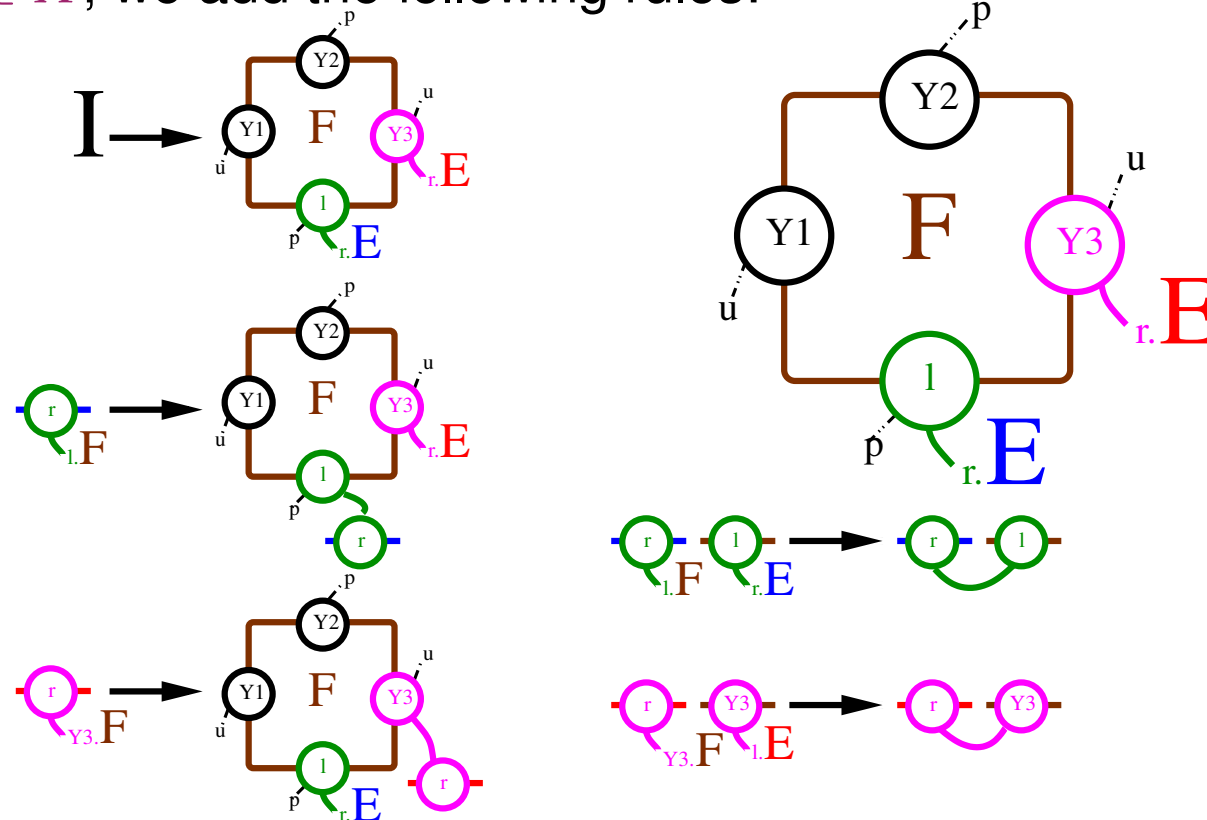
Then:

we have  $\alpha(\{C_i \mid i \in I\}) = \alpha(\{D_j \mid j \in J\})$  (because  $(C_i) \overset{\text{SWAP}}{\sim} (D_j)$ )  
and  $\alpha(\{C_i \mid i \in I\}) \subseteq \alpha(X)$  (because  $(C_i) \in X^*$  and  $\alpha$  mon);  
so  $\alpha(\{D_j \mid j \in J\}) \subseteq \alpha(X)$ ;  
so  $\{D_j \mid j \in J\} \subseteq \gamma(\alpha(X))$  (by def. of Galois connections);  
so  $\{D_j \mid j \in J\} \subseteq X$  (since  $X = \gamma(\alpha(X))$ );  
so  $(D_j)_{j \in J} \in X^*$ .



# Proof: more difficult implication way

For any  $X \in \wp(\text{Local\_view})$ ,  $\gamma(X)$  is given by a rewrite system:  
 For any  $lv \in X$ , we add the following rules:



$I$  and semi-links are non-terminal.  
 $I$  is the initial symbol.

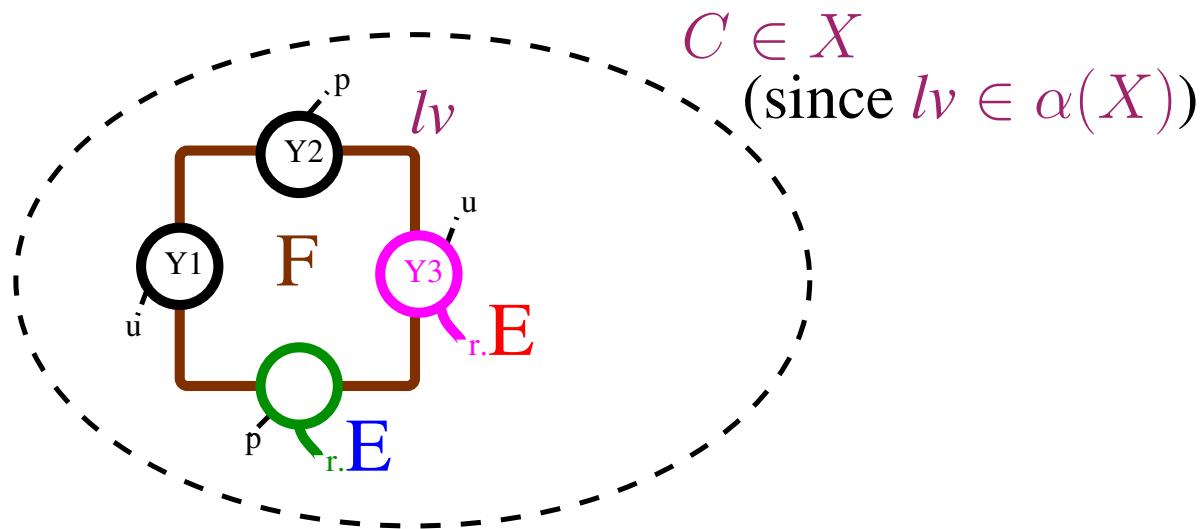
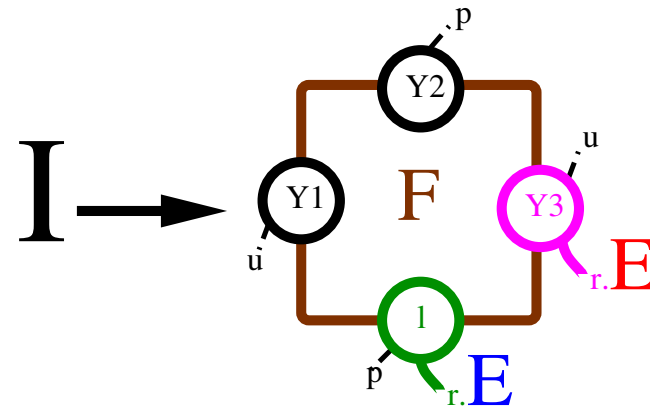
# Proof (more difficult implication way)

We suppose that  $X$  is close with respect to  $\overset{\text{SWAP}}{\sim}$ .  
We want to prove that  $\gamma(\alpha(X)) \subseteq X$ .

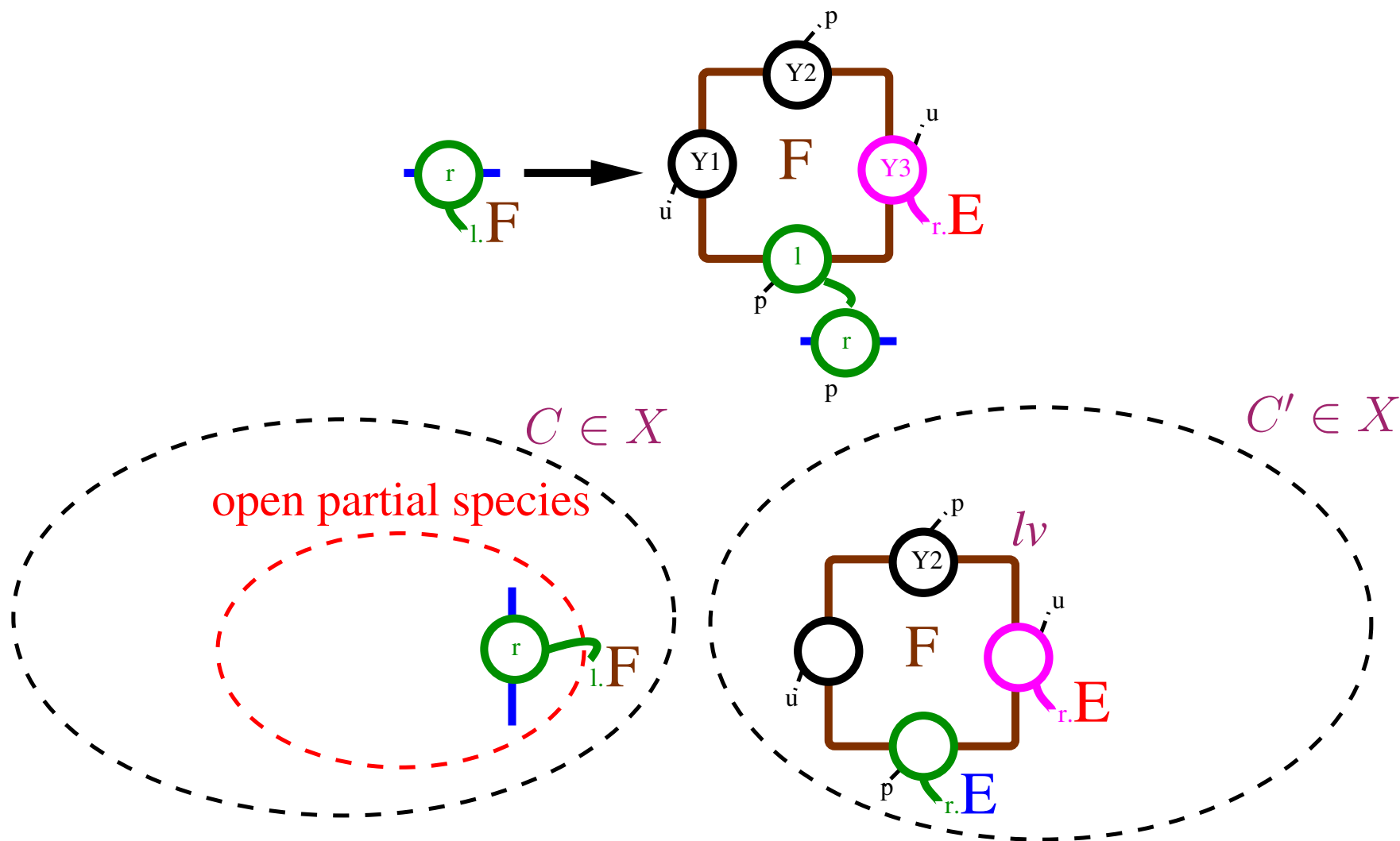
We prove, by induction, that any open complex that can be built by gathering the views of  $\alpha(X)$ , can be embedded in a complex in  $X$ :

- By def. of  $\alpha$ , this is satisfied for any local view in  $\alpha(X)$ ;
- This remains satisfied after unfolding a semi-link with a local view;
- This remains satisfied after binding two semi-links.

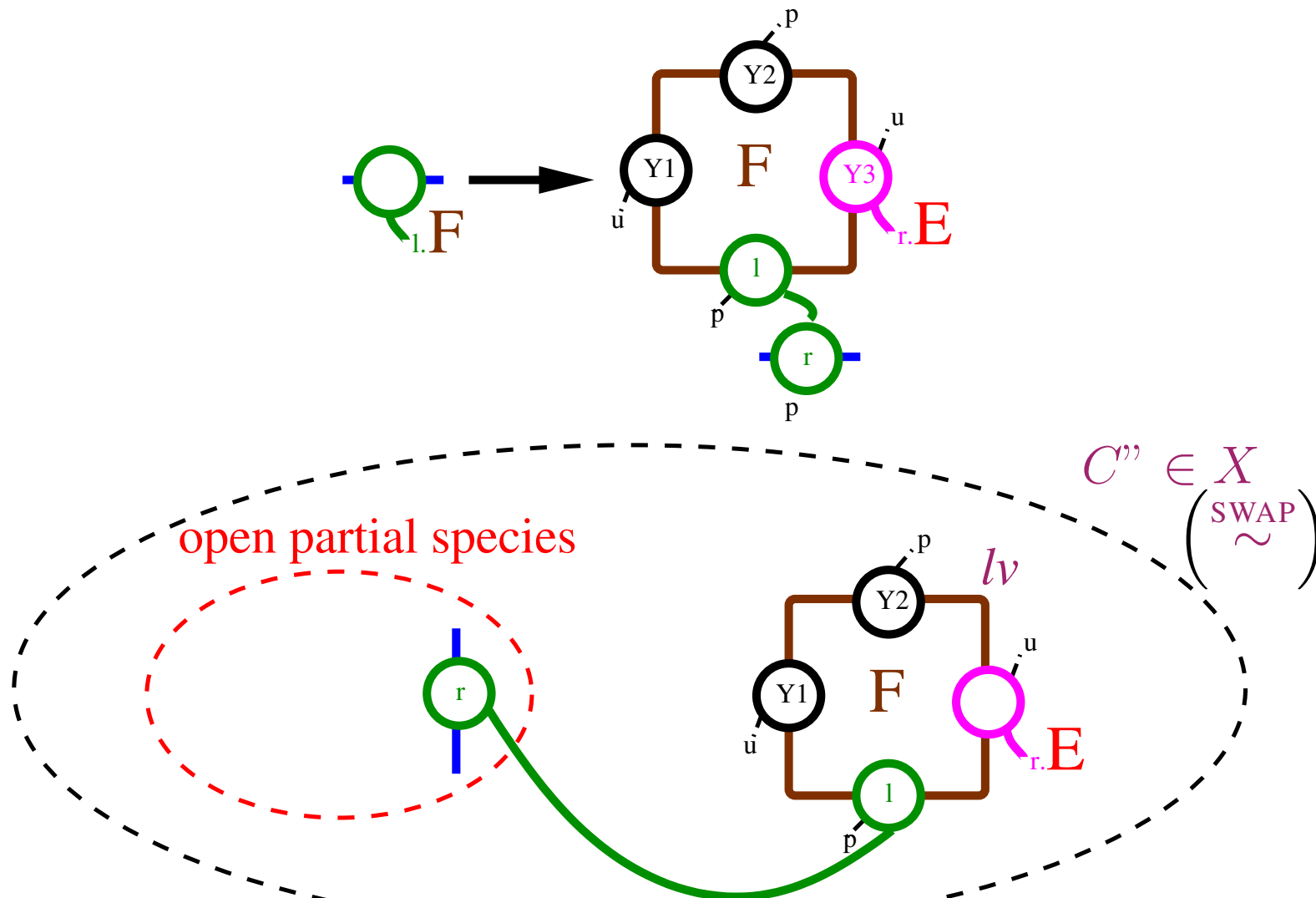
# Initialization



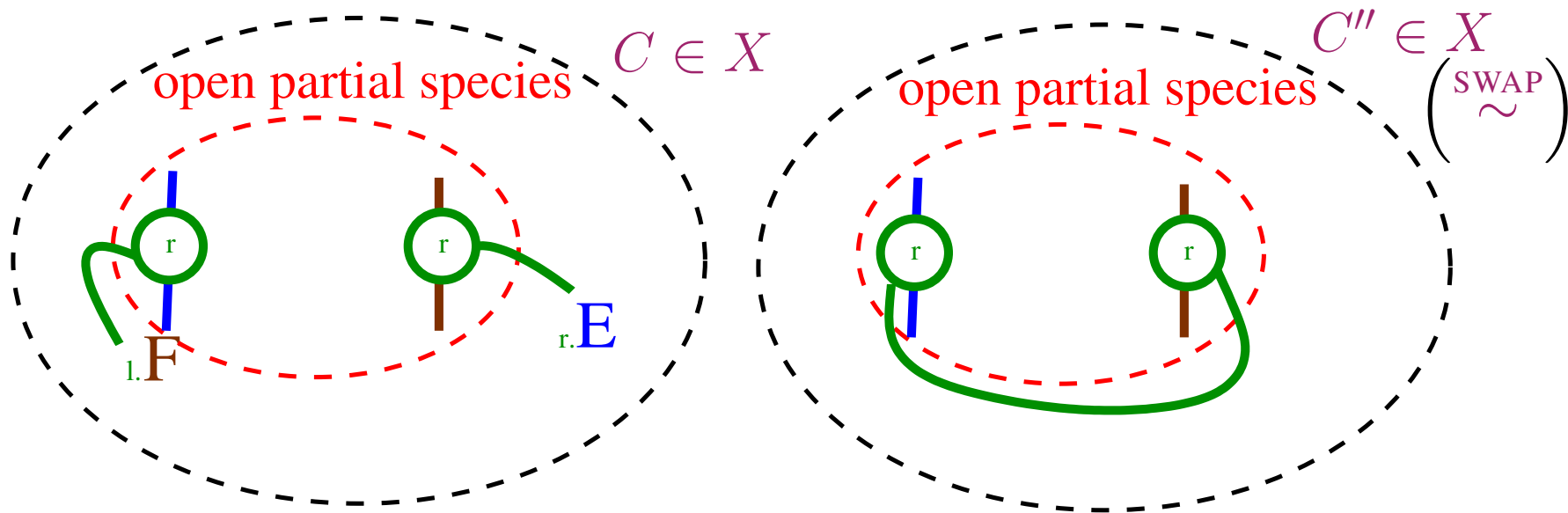
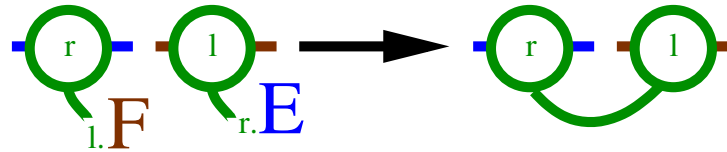
# Unfolding a semi-link



# Unfolding a semi-link



# Binding two semi-links



# Consequences

Let  $Y \in \wp(\text{Local\_view})$  be a set of local views such that  $\alpha(\gamma(Y)) = Y$ .

1. Each open complex  $C$  built with the local views in  $Y$  is a sub-complex of a close complex  $C'$  in  $\gamma(Y)$ .  
(by replacing  $X$  by  $\gamma(Y)$  in the previous proof)
2. When considering the rewrite system that computes  $\gamma(Y)$ , **any partial rewriting sequence can be completed in a successful one.**

Thus:

- (a)  $\gamma(Y)$  is finite if and only if the grammar has a finite set of prefixes (and the latter is decidable);
- (b) We have  $\mathbb{F}^\# \circ \alpha = \alpha \circ \mathbb{F} \circ \gamma \circ \alpha$ .

# Overview

1. Introduction
2. Abstraction: Local views
3. Completeness: false positives?
4. Local fragment of Kappa
5. Other abstract domains
6. Decontextualization
7. Conclusion



# Outline

We have proved that:

- if the set  $Species_\omega$  of reachable chemical species is close with respect to swapping  $\overset{SWAP}{\sim}$ ,
- then the reachability analysis is exact (i.e.  $Species_\omega = \gamma(lfp_{\alpha(Species_0)} \mathbb{F}^\#)$ ).

Now we give some sufficient conditions that ensure this property.

# Sufficient conditions

Whenever the following assumptions:

1. initial agents are not bound;
2. rules are atomic;
3. rules are local:
  - only agents that interact are tested,
  - no cyclic patterns (neither in lhs, nor in rhs);
4. binding rules do not interfere i.e. if both:
  - $A(a\{m\}[\cdot], S), B(b\{n\}[\cdot], T) \rightarrow A(a\{m\}[1], S), B(b\{n\}[1], T)$
  - and  $A(a\{m'\}[\cdot], S'), B(b\{n'\}[\cdot], T') \rightarrow A(a\{m'\}[1], S'), B(b\{n'\}[1], T')$ ,

then:

- $A(a\{m\}[\cdot], S), B(b\{n'\}[\cdot], T') \rightarrow A(a\{m\}[1], S), B(b\{n'\}[1], T')$ ;

5. chemical species in  $\gamma(\alpha(\textit{Species}_\omega))$  are acyclic,

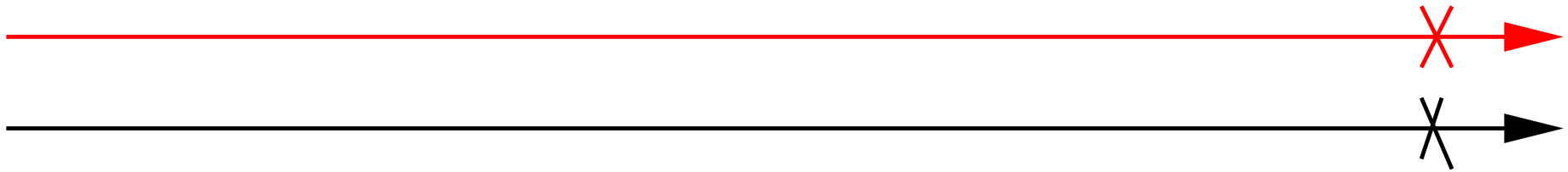
are satisfied, the set of reachable chemical species is local.

# Proof outline

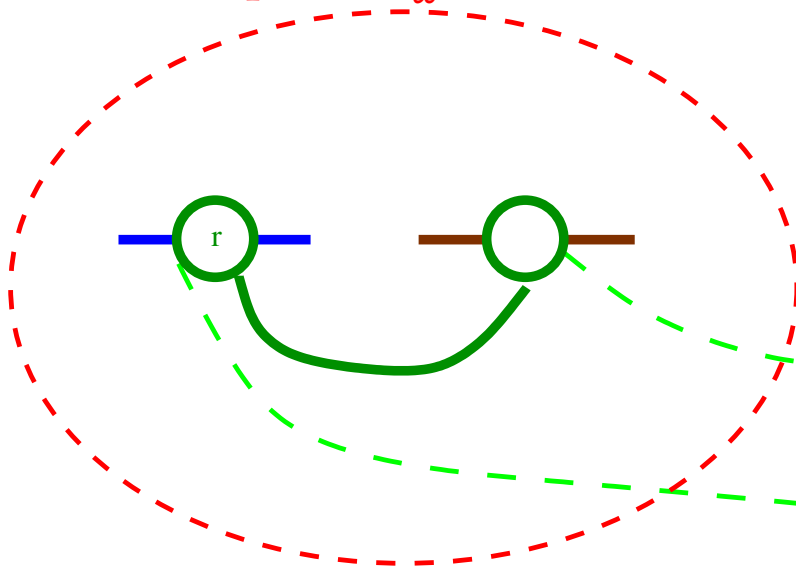
We sketch a proof in order to discover sufficient conditions that ensure this property:

- We consider tuples of complexes in which the same kind of links occur twice.
- We want to swap these links.
- We introduce the history of their computation.
- There are several cases. . .

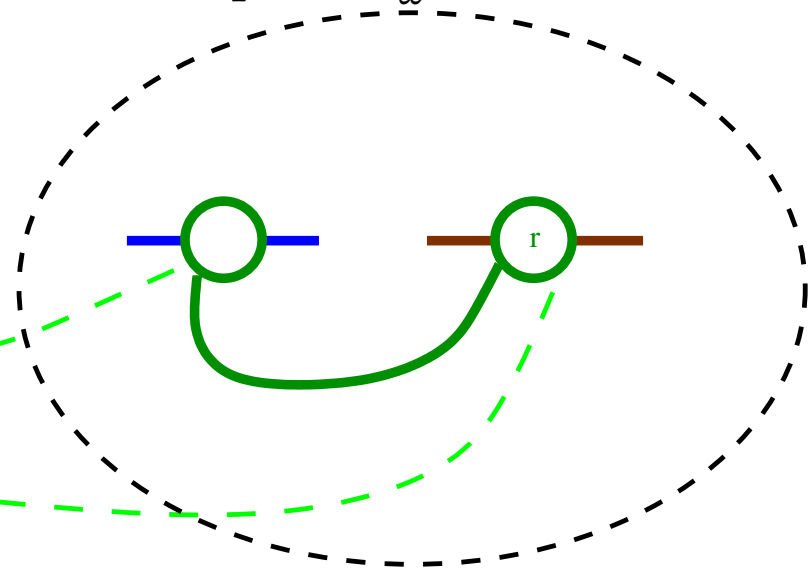
# First case (I/V)



$C \in \text{Species}_\omega$

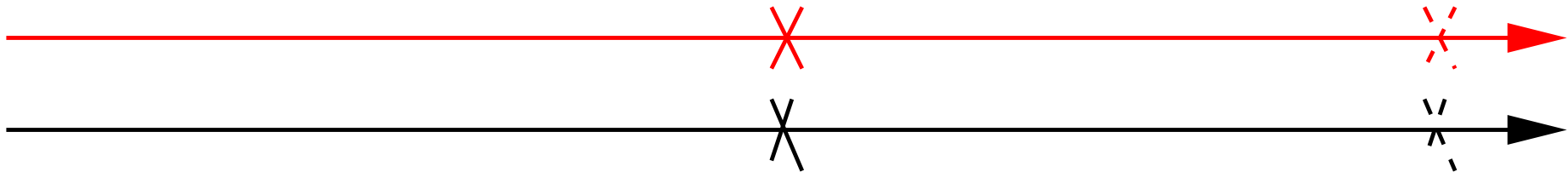


$C' \in \text{Species}_\omega$

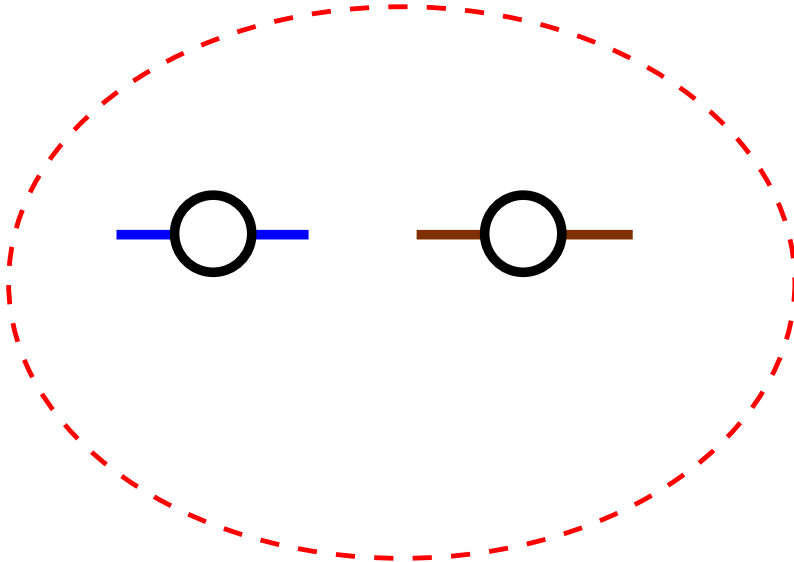


# First case (II/V)

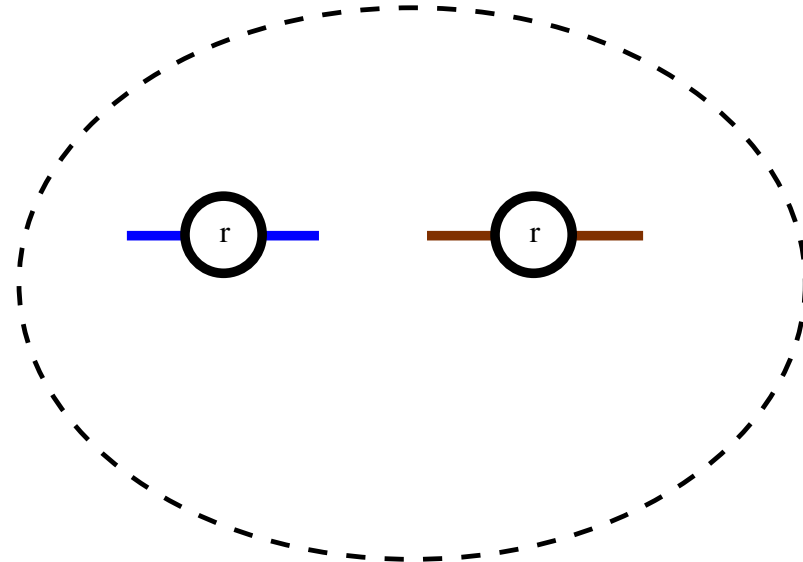
just before the links are made



$C \in \text{Species}_{\omega}^*$

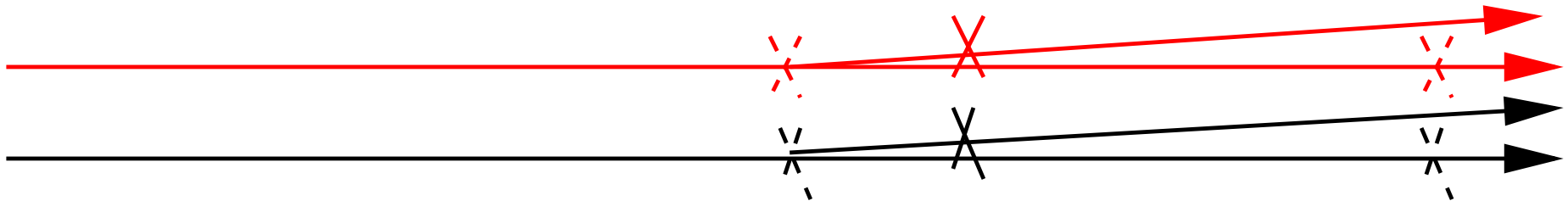


$C' \in \text{Species}_{\omega}^*$

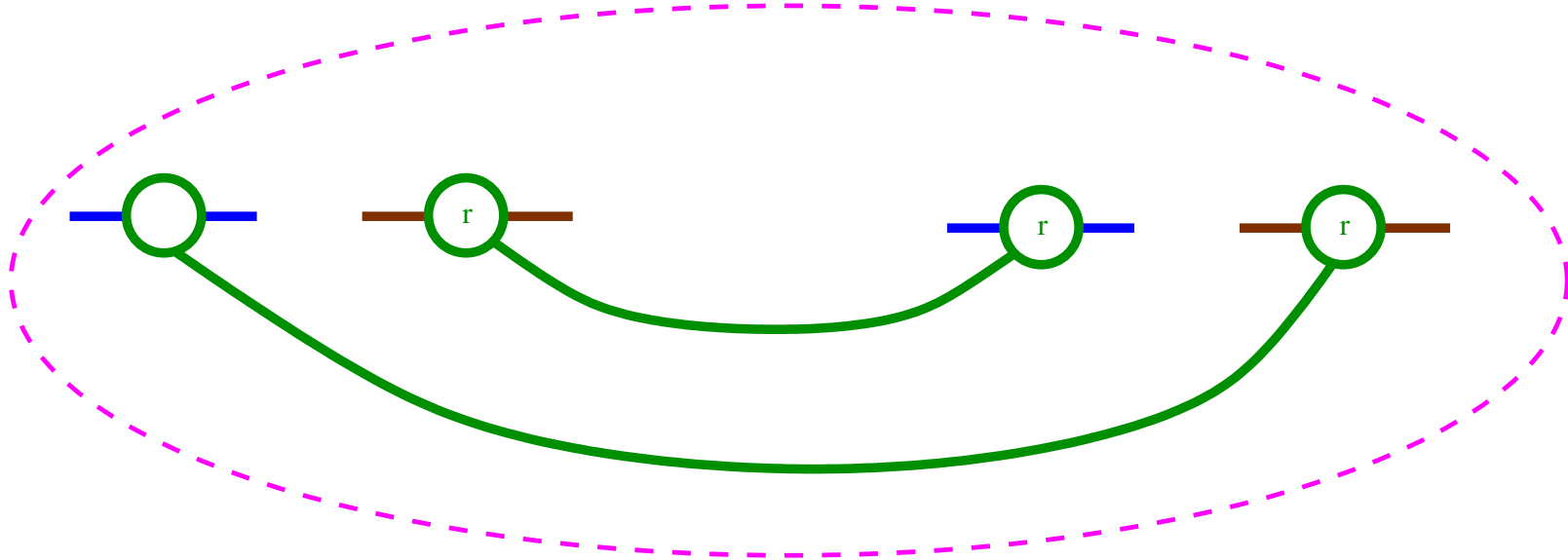


# First case (III/V)

we suppose we can swap the links



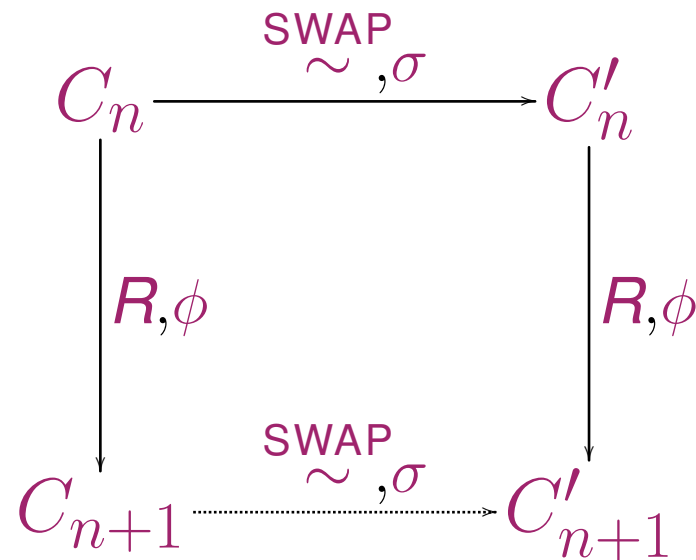
$C \in \text{Species}_\omega^*$



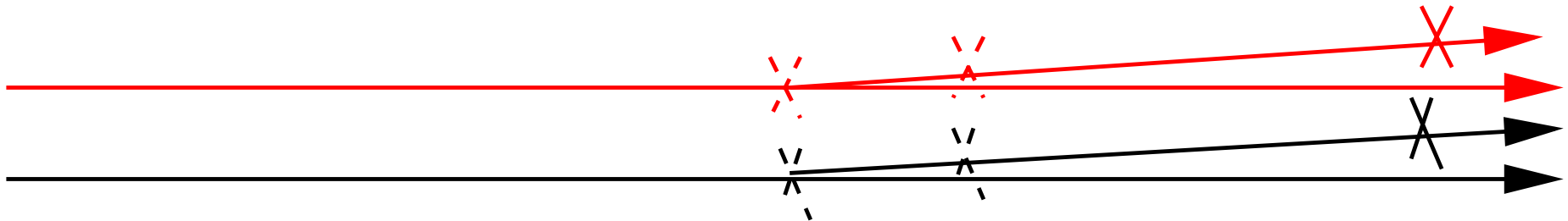
# First case (IV/V)

Then, we ensure that further computation steps:

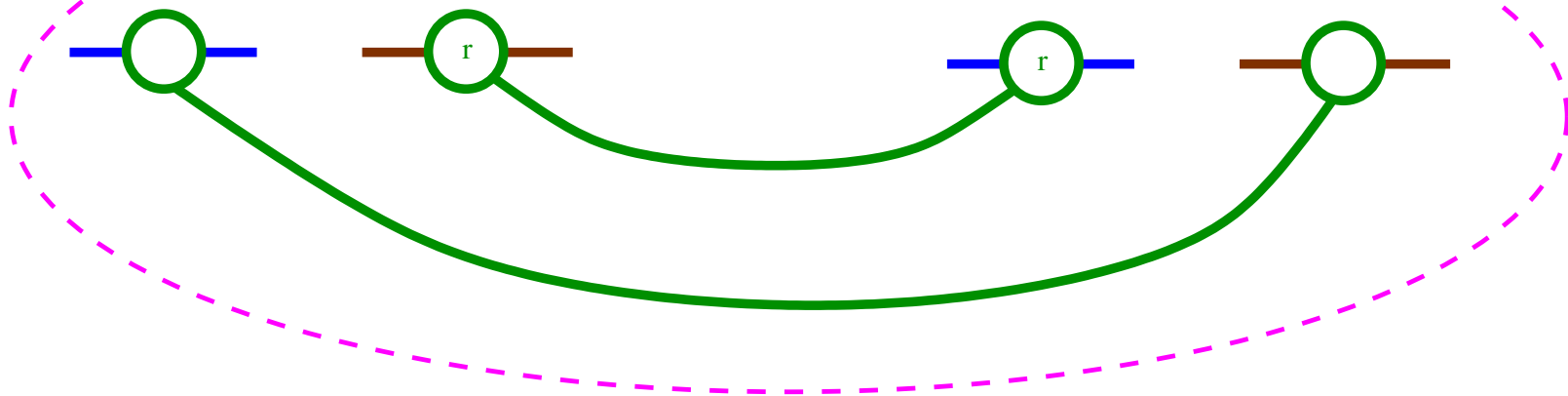
- are always possible;
- have the same effect on local views;
- commute with the swapping relation  $\overset{\text{SWAP}}{\sim}$ .



# First case (V/V)



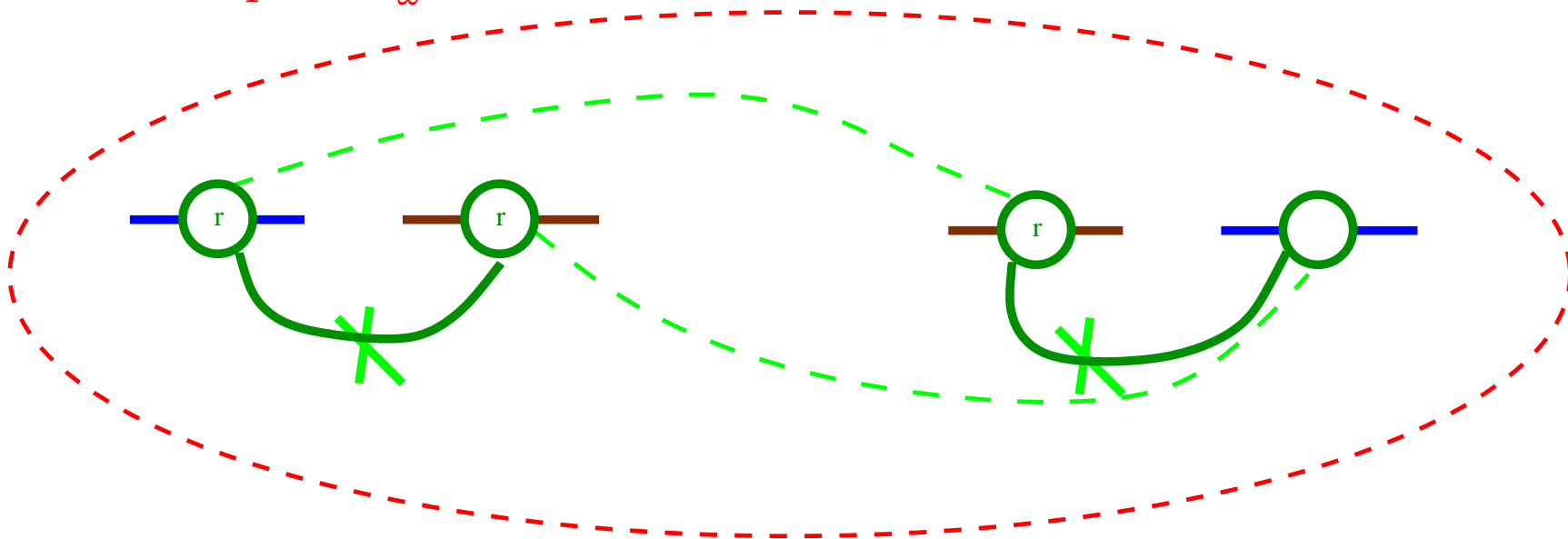
$C \in \text{Species}_\omega^*$





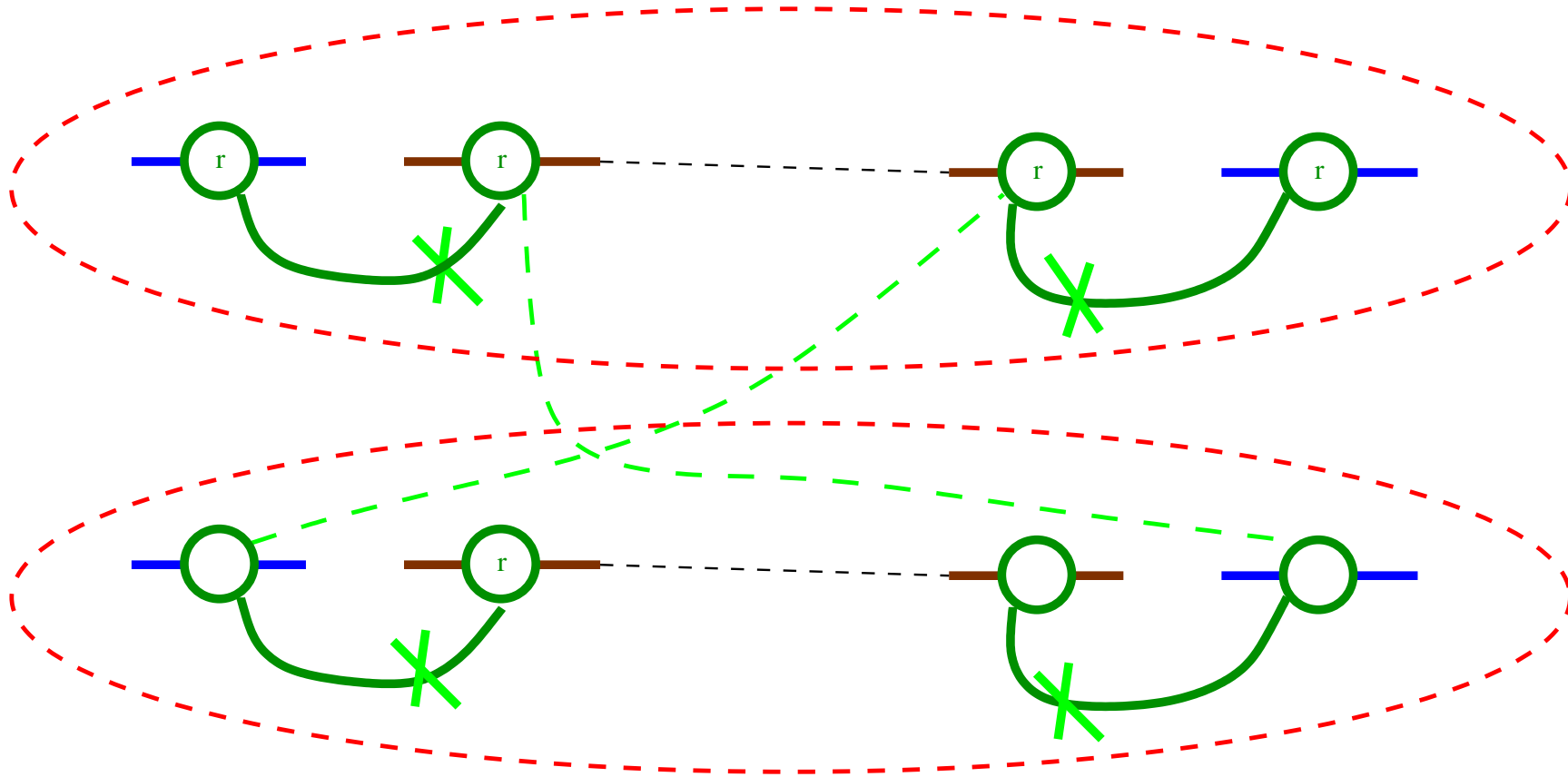
## Second case (I/II)

$C \in \text{Species}_\omega$



we assume that the chemical species  $C$  is acyclic

## Second case (II/II)



# Sufficient conditions

Whenever the following assumptions:

1. initial agents are not bound;
2. rules are atomic;
3. rules are local:
  - only agents that interact are tested,
  - no cyclic patterns (neither in lhs, nor in rhs);
4. binding rules do not interfere i.e. if both:
  - $A(a\{m\}[\cdot], S), B(b\{n\}[\cdot], T) \rightarrow A(a\{m\}[1], S), B(b\{n\}[1], T)$
  - and  $A(a\{m'\}[\cdot], S'), B(b\{n'\}[\cdot], T') \rightarrow A(a\{m'\}[1], S'), B(b\{n'\}[1], T')$ ,

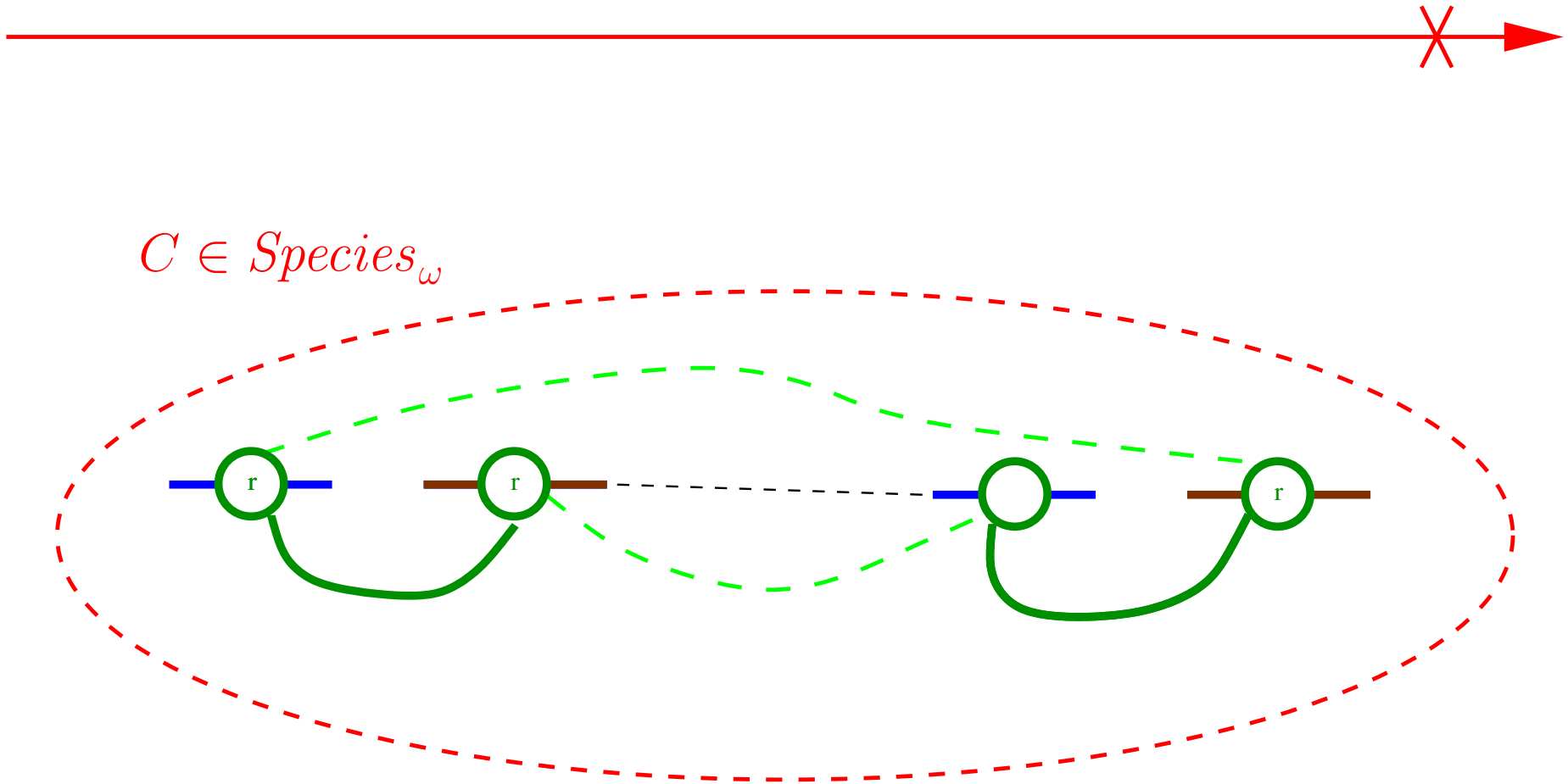
then:

- $A(a\{m\}[\cdot], S), B(b\{n'\}[\cdot], T') \rightarrow A(a\{m\}[1], S), B(b\{n'\}[1], T')$ ;

5. chemical species in  $\gamma(\alpha(\textit{Species}_\omega))$  are acyclic,

are satisfied, the set of reachable chemical species is local.

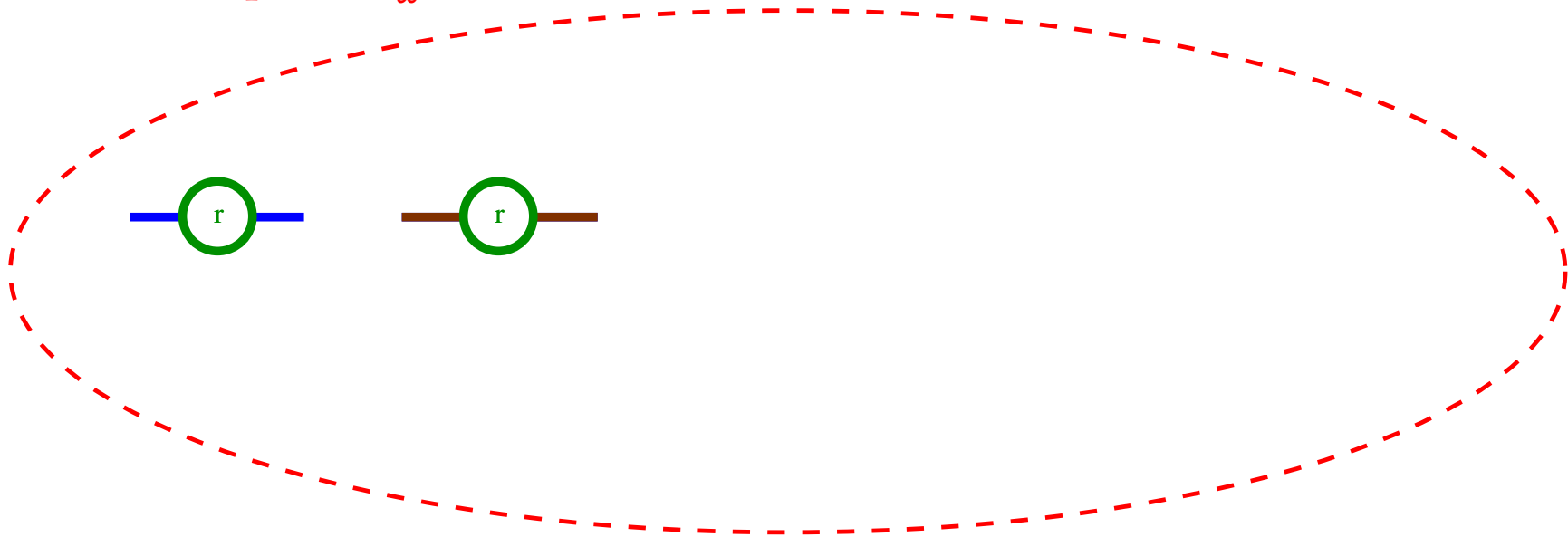
# Third case (I/III)



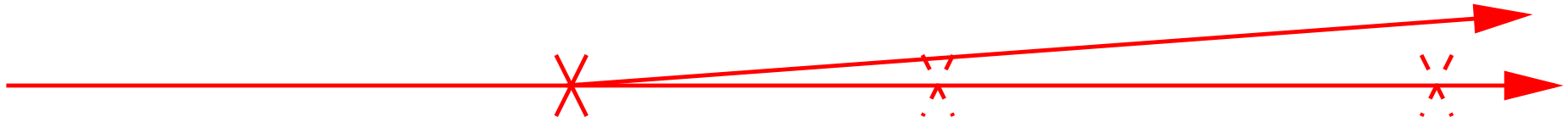
## Third case (II/III)



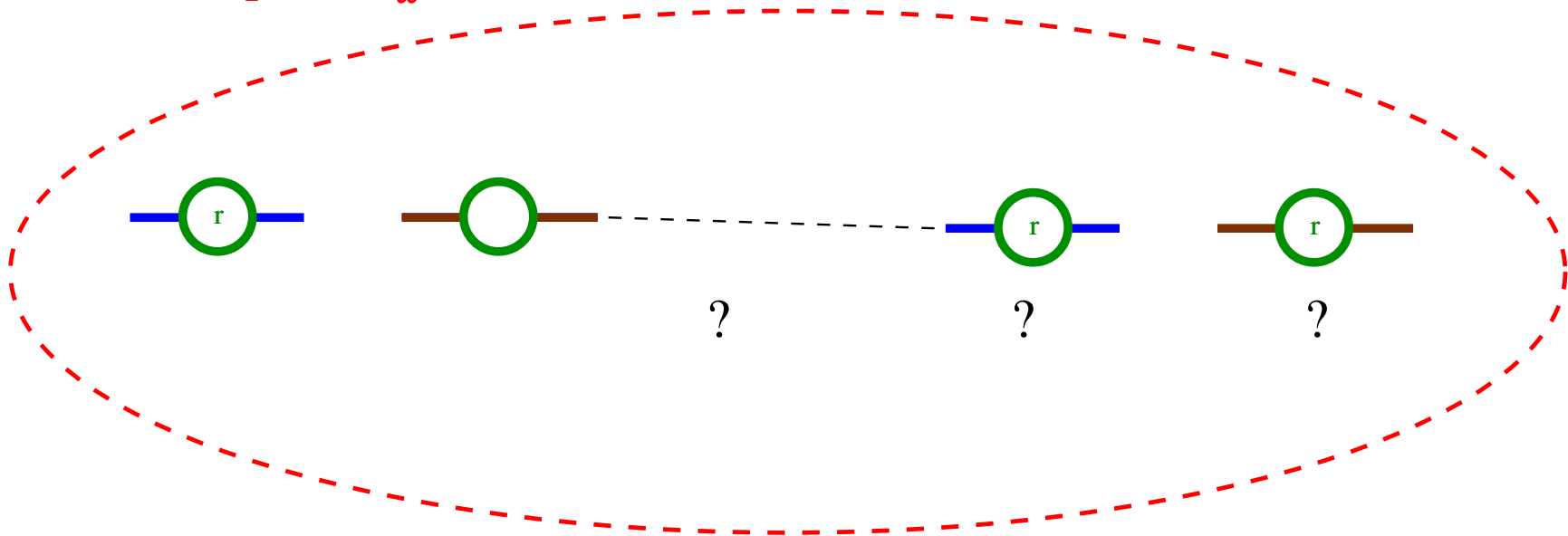
$C \in \text{Species}_{\omega}^*$



## Third case (II/III)



$C \in \text{Species}_{\omega}^*$



# Dangerous sites

A site is dangerous if it may occur in a cycle within a complex ( $\in \gamma(\alpha(\textit{Species}_\omega))$ ).

We would weaken the fifth requirement into:

- The binding state of a dangerous site is never tested, unless for binding or unbinding this site.
- When we bind dangerous sites, we only test that these sites are free.

Then, we prove that:

1. we can build any complex with free dangerous sites,
2. then, we can bind them as much as we like.

# Non local systems

$$Species_0 \triangleq R(a\{u\}[\cdot])$$

$$Rules \triangleq \left\{ \begin{array}{l} R(a\{u\}[\cdot]) \leftrightarrow R(a\{p\}[\cdot]) \\ R(a\{u\}[\cdot]), R(a\{u\}[\cdot]) \rightarrow R(a\{u\}[1]), R(a\{u\}[1]) \\ R(a\{p\}[\cdot]), R(a\{u\}[\cdot]) \rightarrow R(a\{p\}[1]), R(a\{p\}[1]) \\ R(a\{p\}[\cdot]), R(a\{p\}[\cdot]) \rightarrow R(a\{p\}[1]), R(a\{p\}[1]) \end{array} \right\}$$

$$R(a\{u\}[1]), R(a\{u\}[1]) \in Species_\omega$$

$$R(a\{p\}[1]), R(a\{p\}[1]) \in Species_\omega$$

$$\text{But } R(a\{u\}[1]), R(a\{p\}[1]) \notin Species_\omega.$$



# Non local systems

$$Species_0 \triangleq A(a\{u\}[\cdot]), B(a\{u\}[\cdot])$$

$$Rules \triangleq \left\{ \begin{array}{l} A(a\{u\}[\cdot]), B(a\{u\}[\cdot]) \rightarrow A(a\{u\}[1]), B(a\{u\}[1]) \\ A(a\{u\}[1]), B(a\{u\}[1]) \rightarrow A(a\{p\}[1]), B(a\{u\}[1]) \\ A(a\{u\}[1]), B(a\{u\}[1]) \rightarrow A(a\{u\}[1]), B(a\{p\}[1]) \end{array} \right\}$$

$$A(a\{u\}[1]), B(a\{p\}[1]) \in Species_\omega$$

$$A(a\{p\}[1]), B(a\{u\}[1]) \in Species_\omega$$

$$\text{But } A(a\{p\}[1]), B(a\{p\}[1]) \notin Species_\omega.$$

# Non local systems

$$\begin{aligned} \textit{Species}_0 &\triangleq A(a\{u\}[\cdot]) \\ \textit{Rules} &\triangleq \left\{ \begin{array}{l} A(a\{u\}[\cdot]) \leftrightarrow A(a\{p\}[\cdot]) \\ A(a\{u\}[\cdot]), A(a\{p\}[\cdot]) \rightarrow A(a\{u\}[1]), A(a\{p\}[1]) \end{array} \right\} \end{aligned}$$

$A(a\{u\}[1]), A(a\{p\}[1]) \in \textit{Species}_\omega$   
**But**  $A(a\{p\}[1]), A(a\{p\}[1]) \notin \textit{Species}_\omega$ .

# Non local systems

*Species*<sub>0</sub>  $\triangleq$  R(a[.],b[.])

*Rules*  $\triangleq$  { R(a[.],b[.]),R(a[.])  $\rightarrow$  R(a[.],b[1]),R(a[1]) }

R(a[.],b[2]),R(a[2],b[1]),R(a[1],b[.])  $\in$  *Species* <sub>$\omega$</sub>

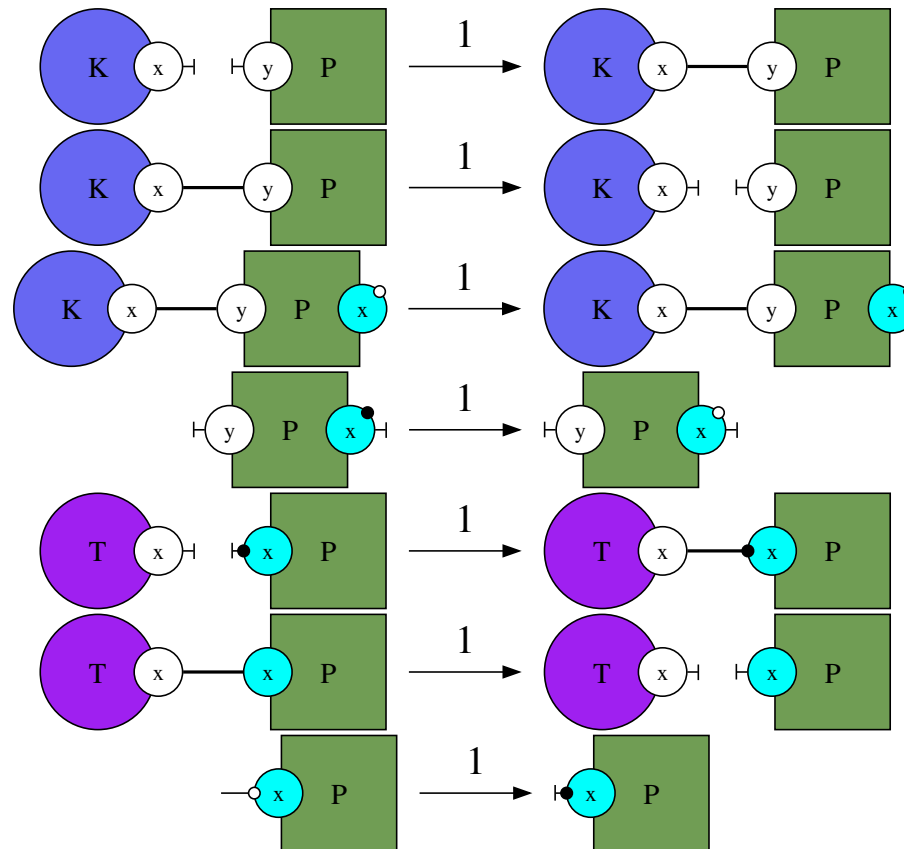
**But** R(a[1],b[1])  $\notin$  *Species* <sub>$\omega$</sub> .

# Overview

1. Introduction
2. Abstraction: Local views
3. Completeness: false positives?
4. Local fragment of Kappa
5. Other abstract domains
6. Decontextualization
7. Conclusion

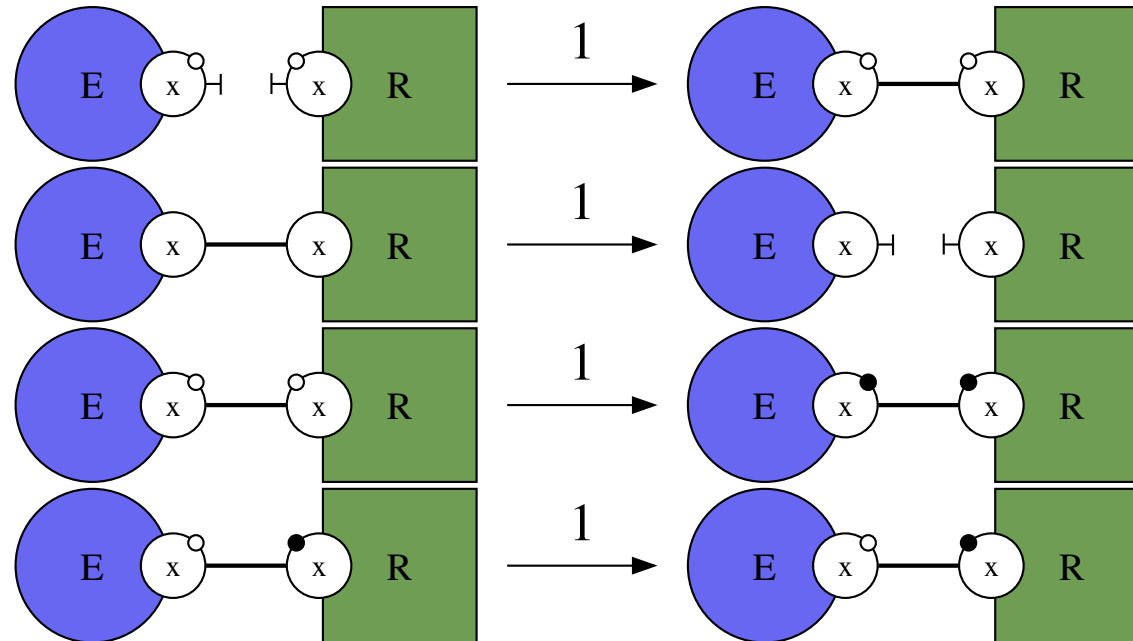
# Practical activities

Use static analysis on the following model:



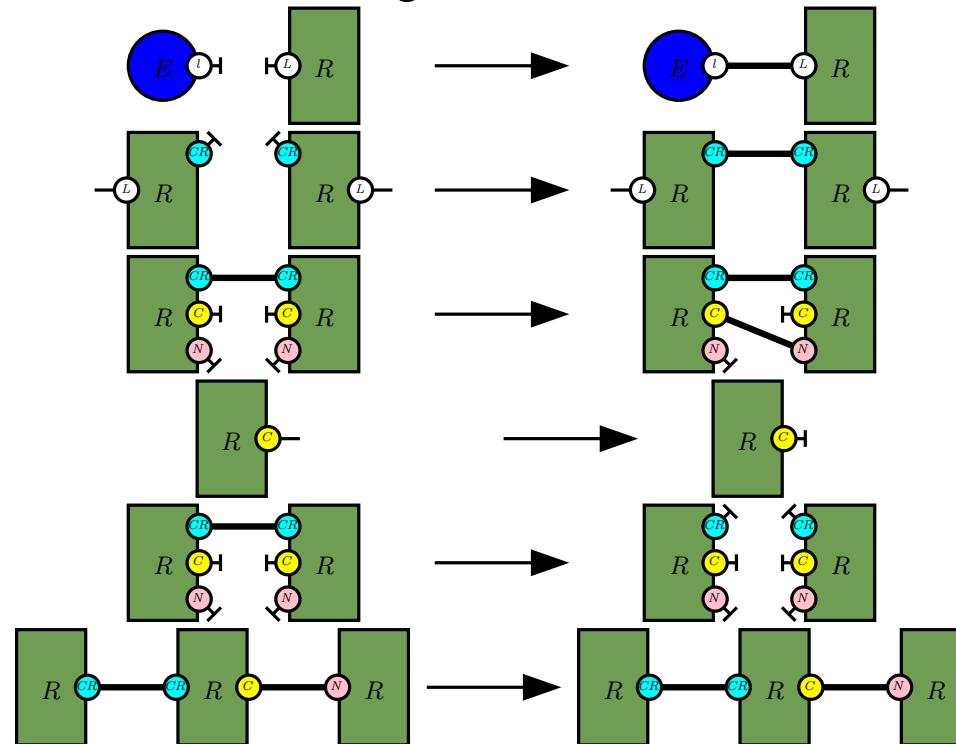
# Practical activities

Use static analysis on the following model:



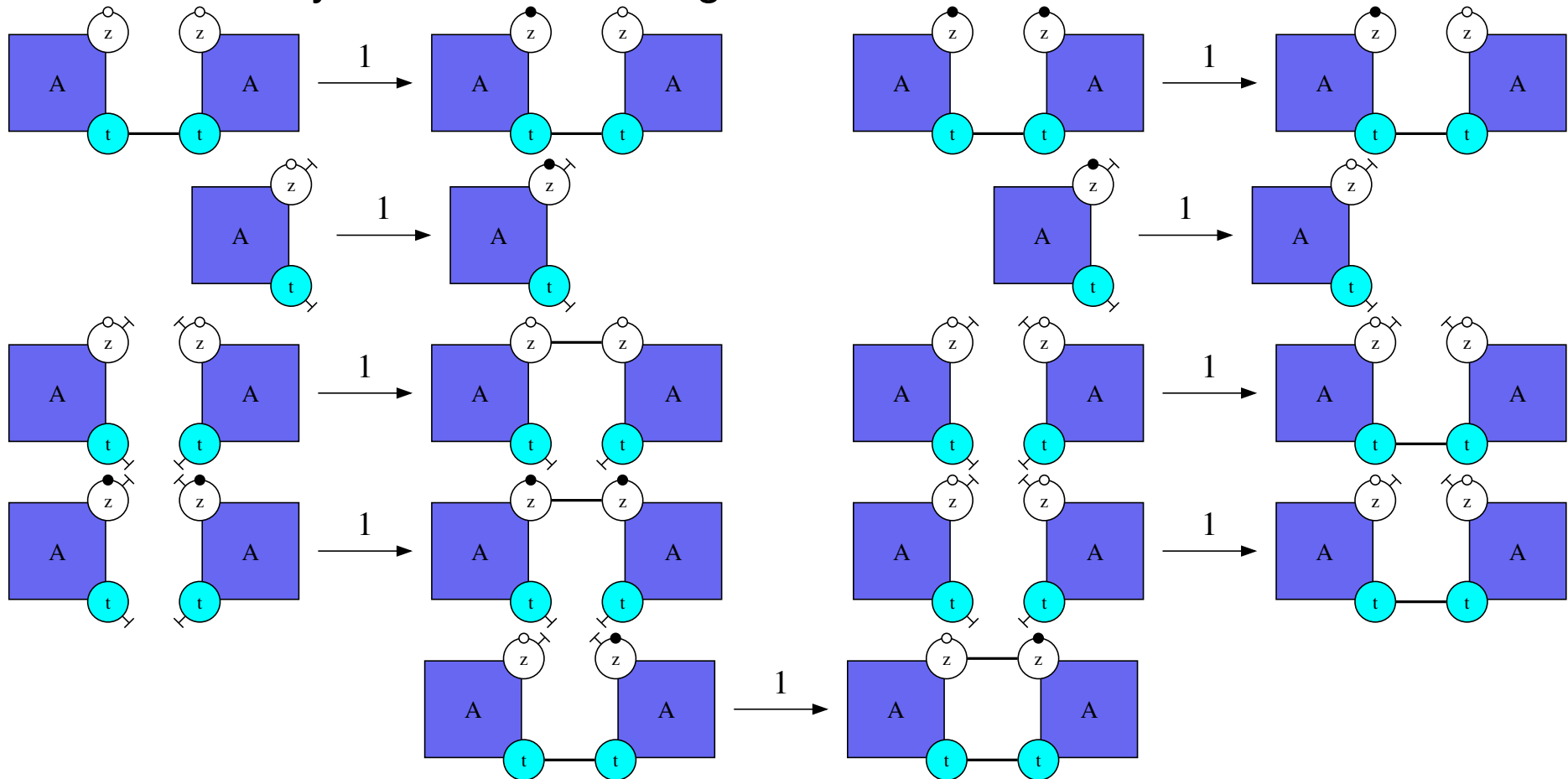
# Practical activities

Use static analysis on the following model:



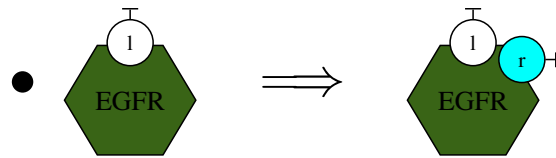
# Practical activities

Use static analysis on the following model:

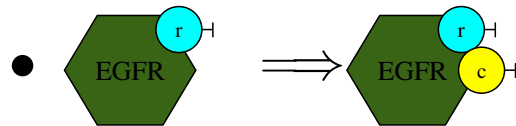




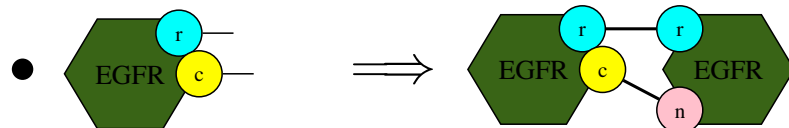
# Some invariants



whenever the site  $l$  is free, the site  $r$  is free as well.



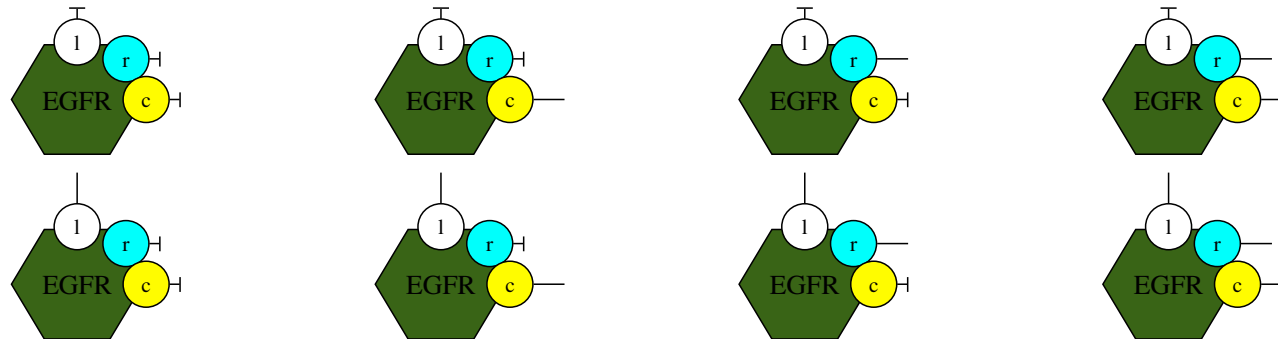
whenever the site  $r$  is free, the site  $c$  is free as well.



whenever the sites  $c$  and  $r$  are both bound, they are bound to a single occurrence of an occurrence of the protein *EGFR*.

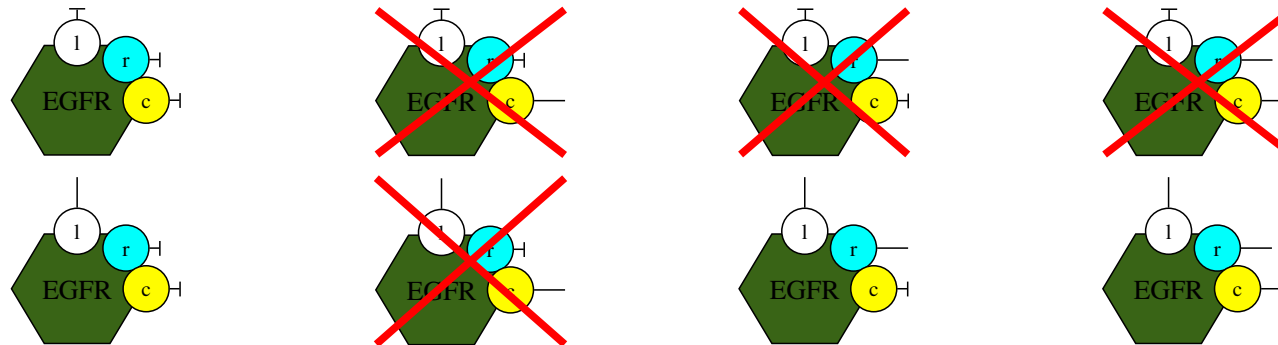
# Abstract domain

Among a set of patterns,



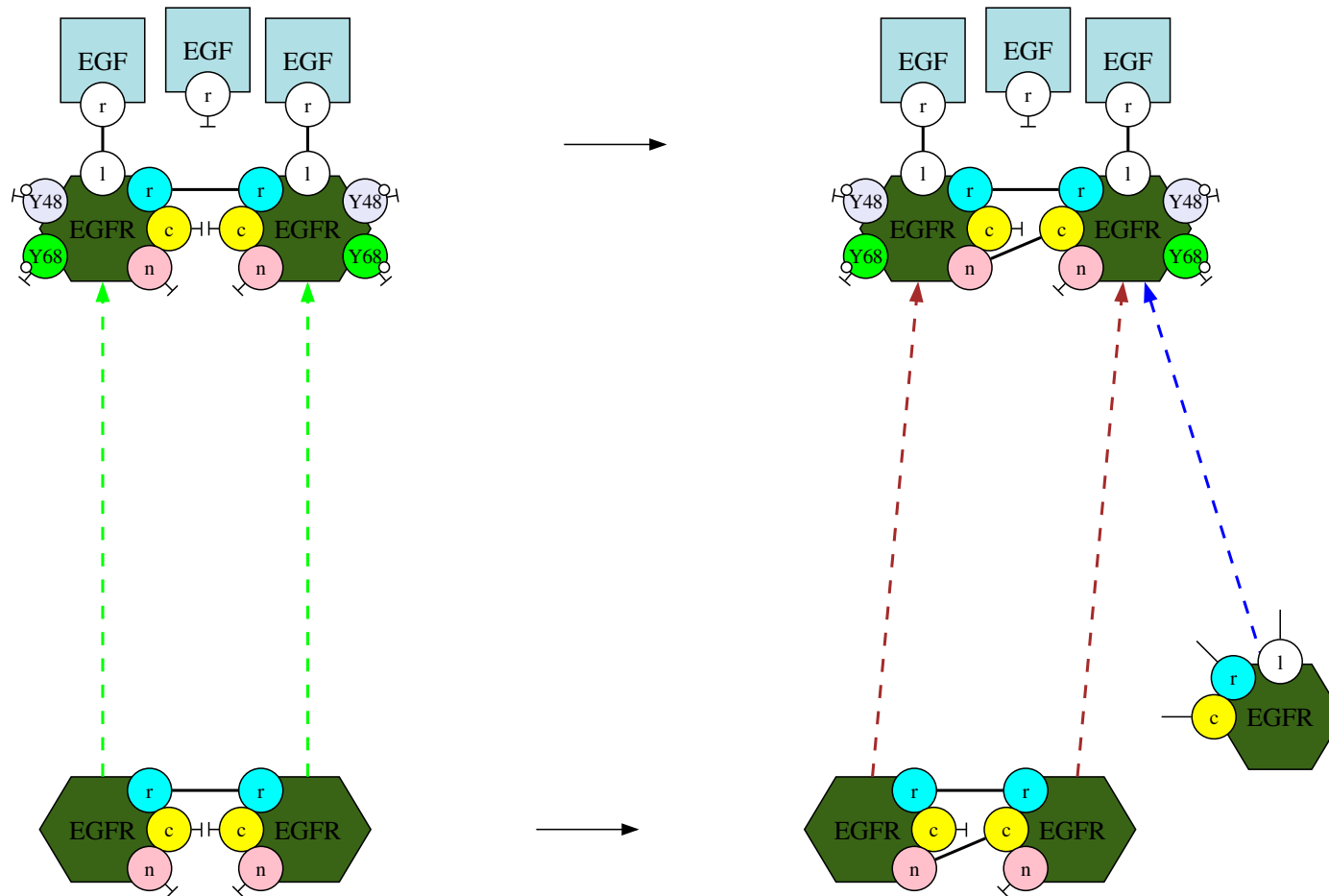
# Abstract domain

Among a set of patterns,

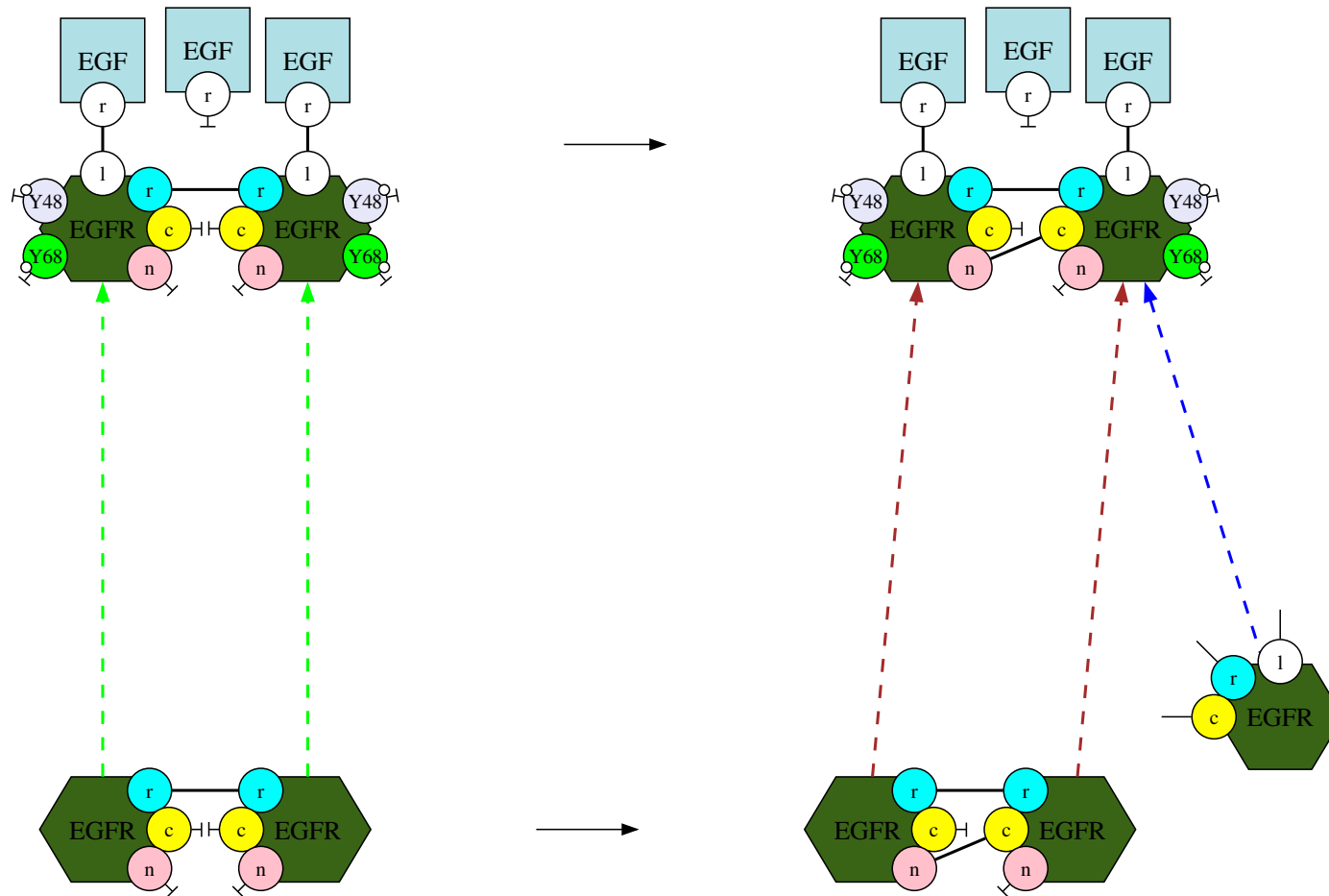


we would like to compute which ones may occur in a reachable state.  
(*the initial state is given in the specification.*)

# Most precise abstract transition

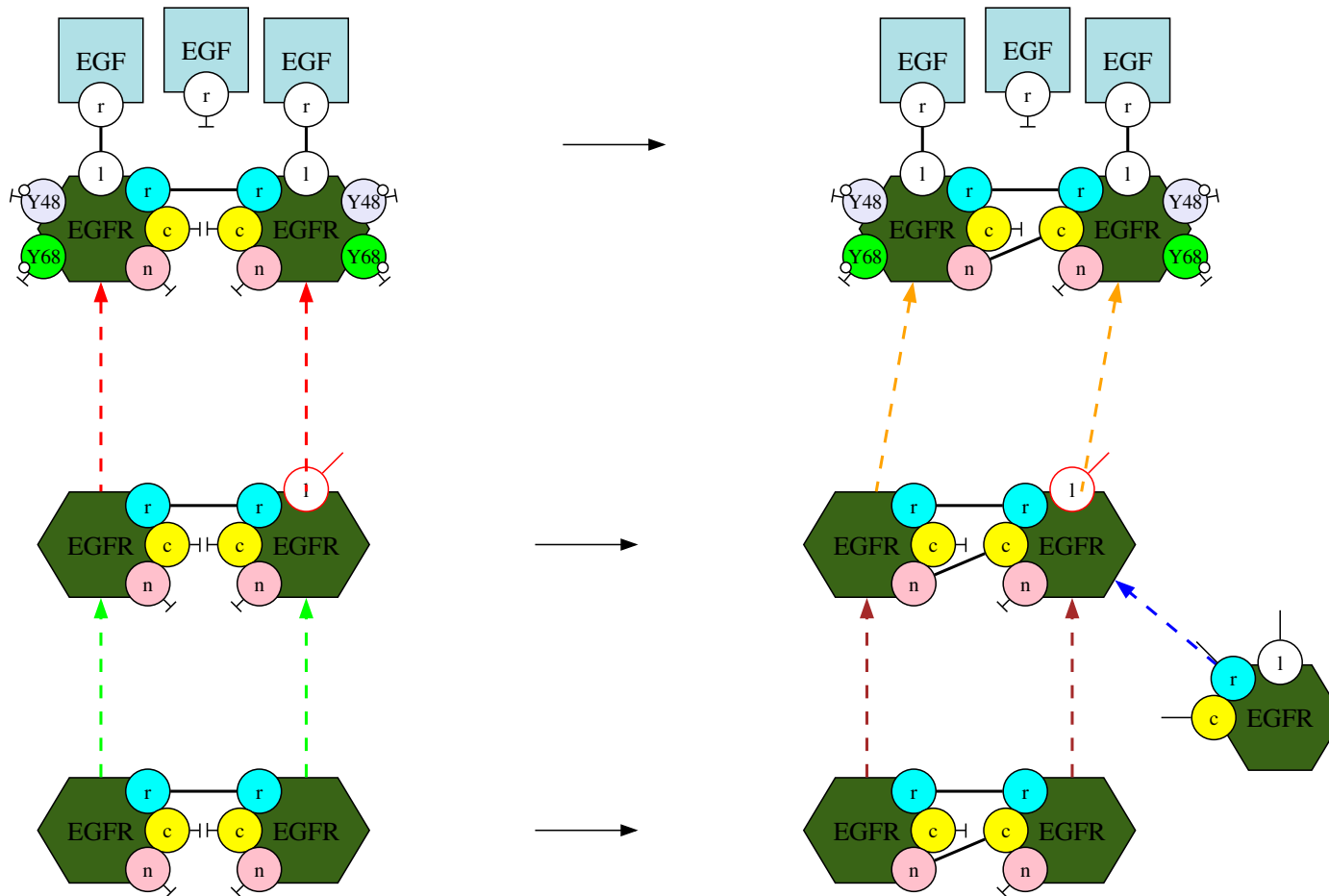


# Most precise abstract transition

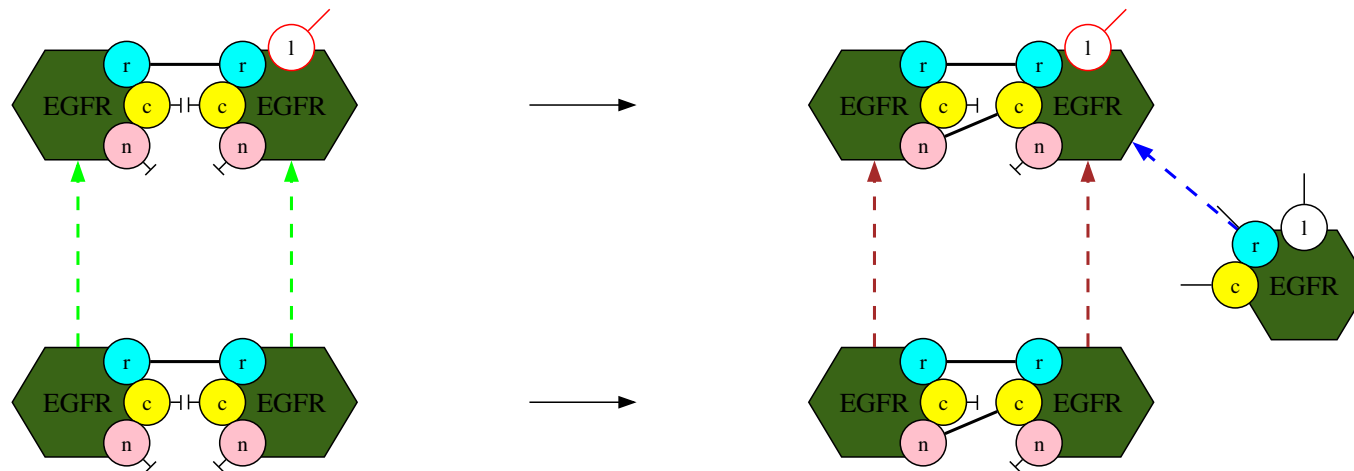


when the left hand side of the transition (top) contains no forbidden pattern.

# Simplified abstract transition

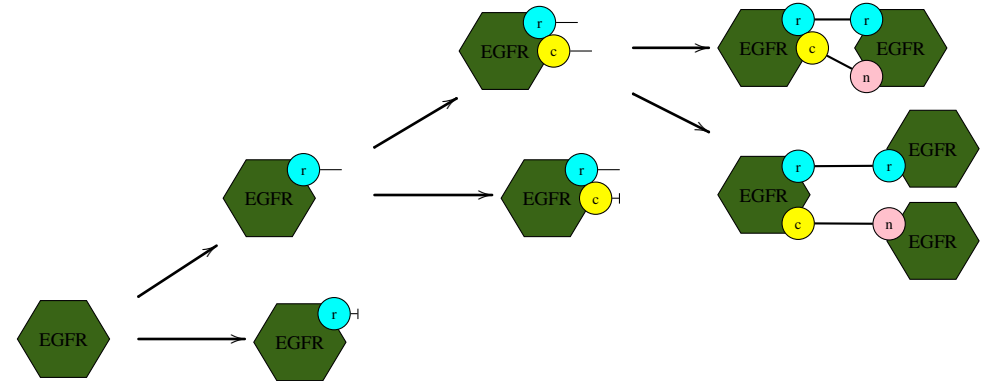
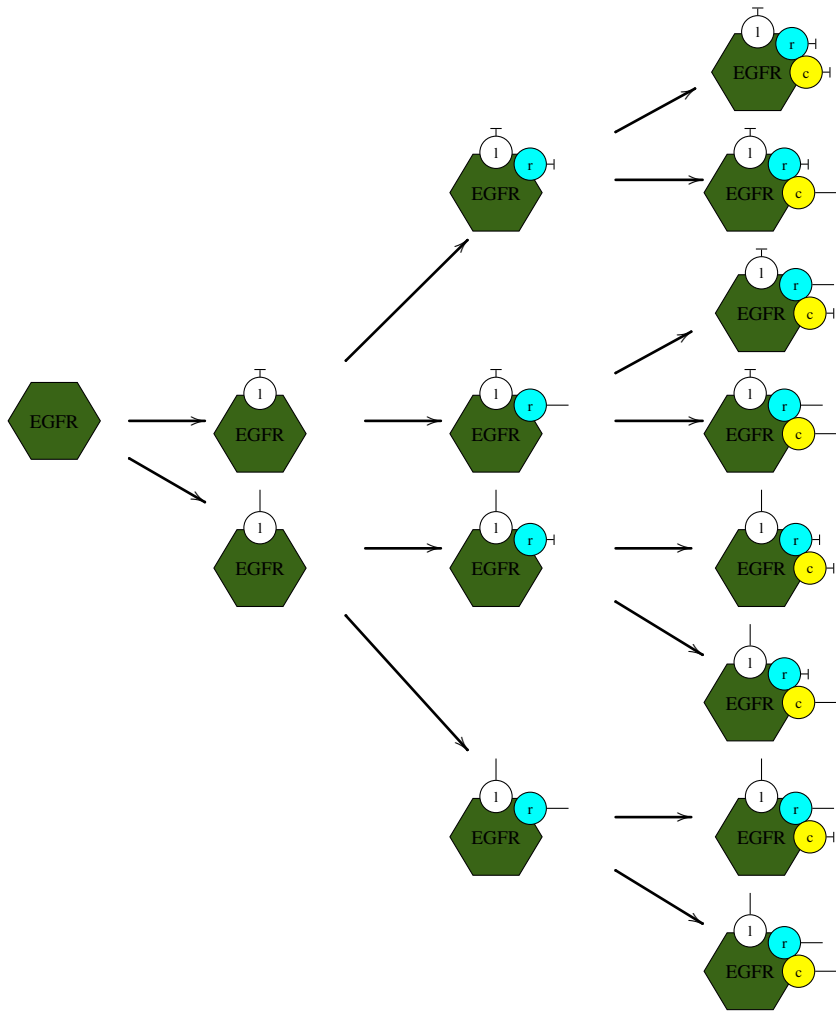


# Simplified abstract transition



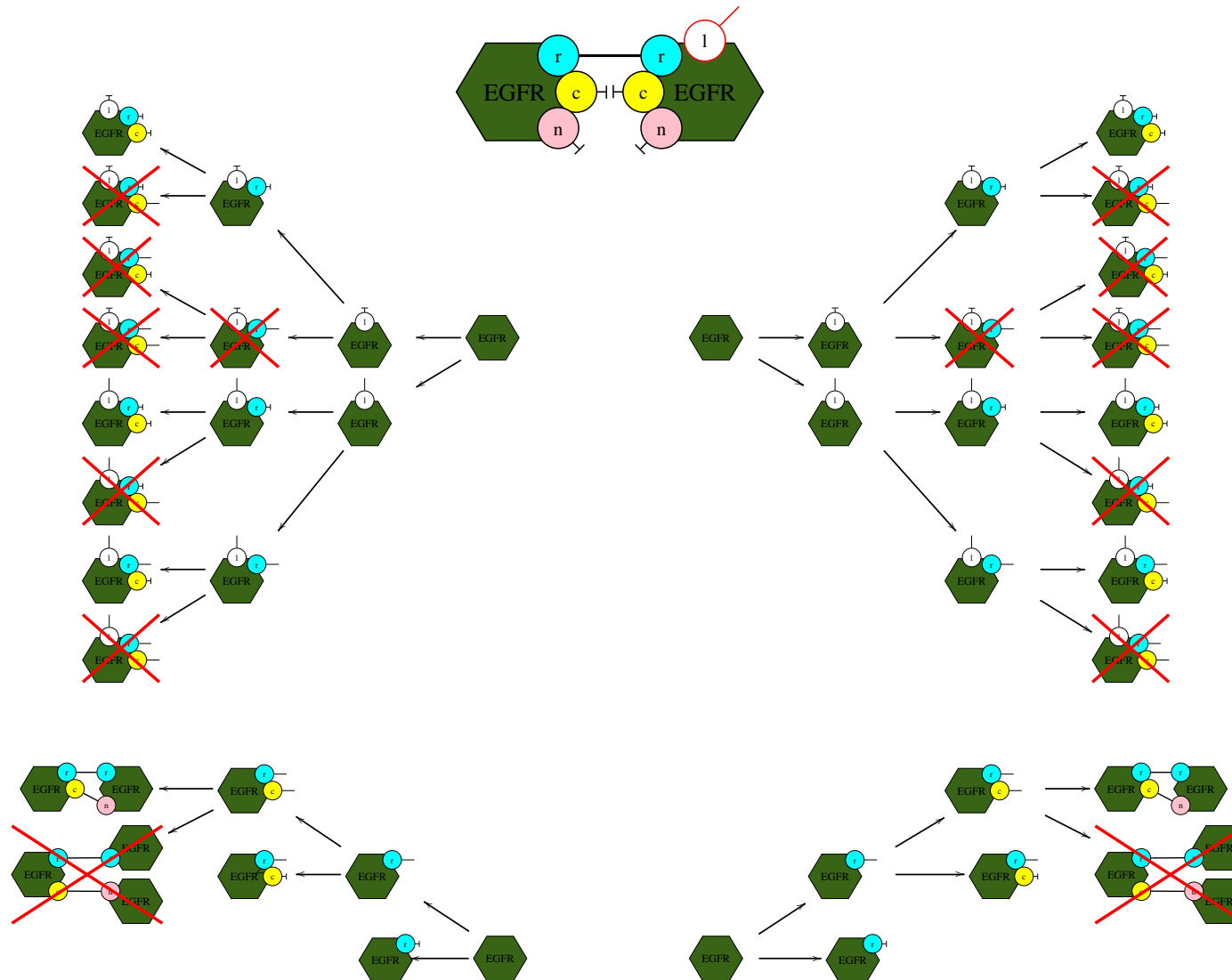
We are left to check whether the left hand side of the refined rule is reachable given the forbidden patterns.

# Orthogonal sets of patterns

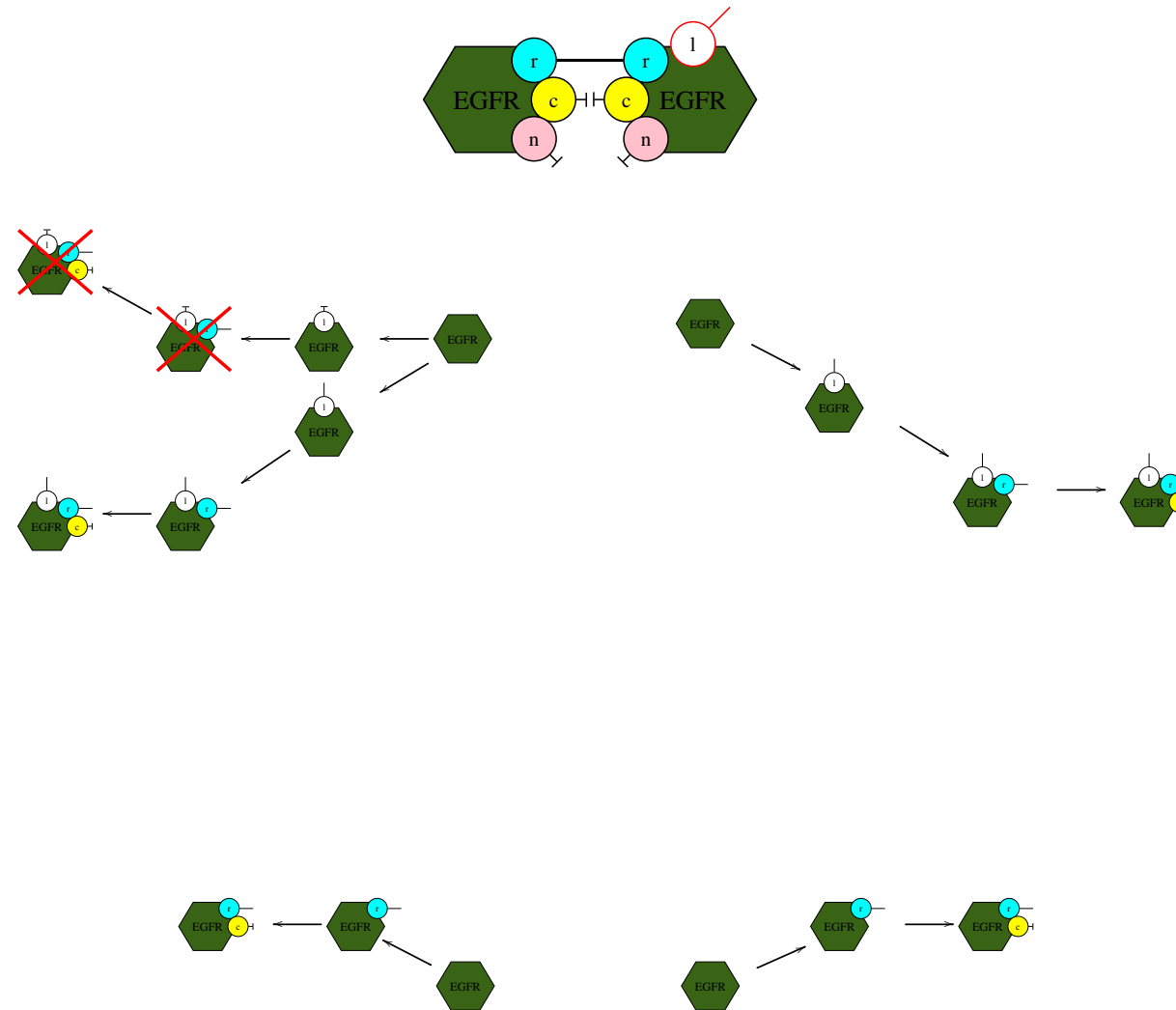




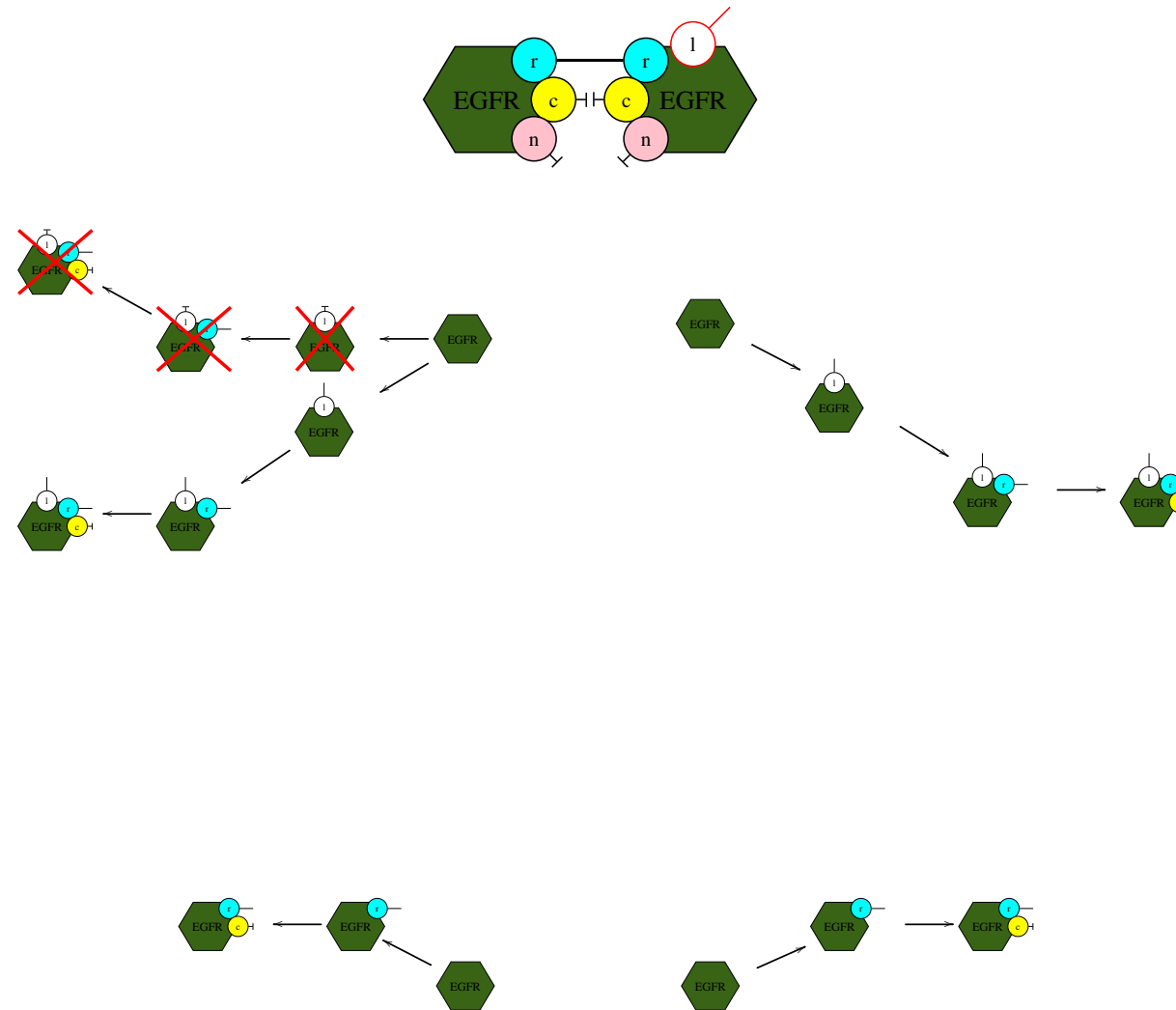
# Compatibility checking procedure



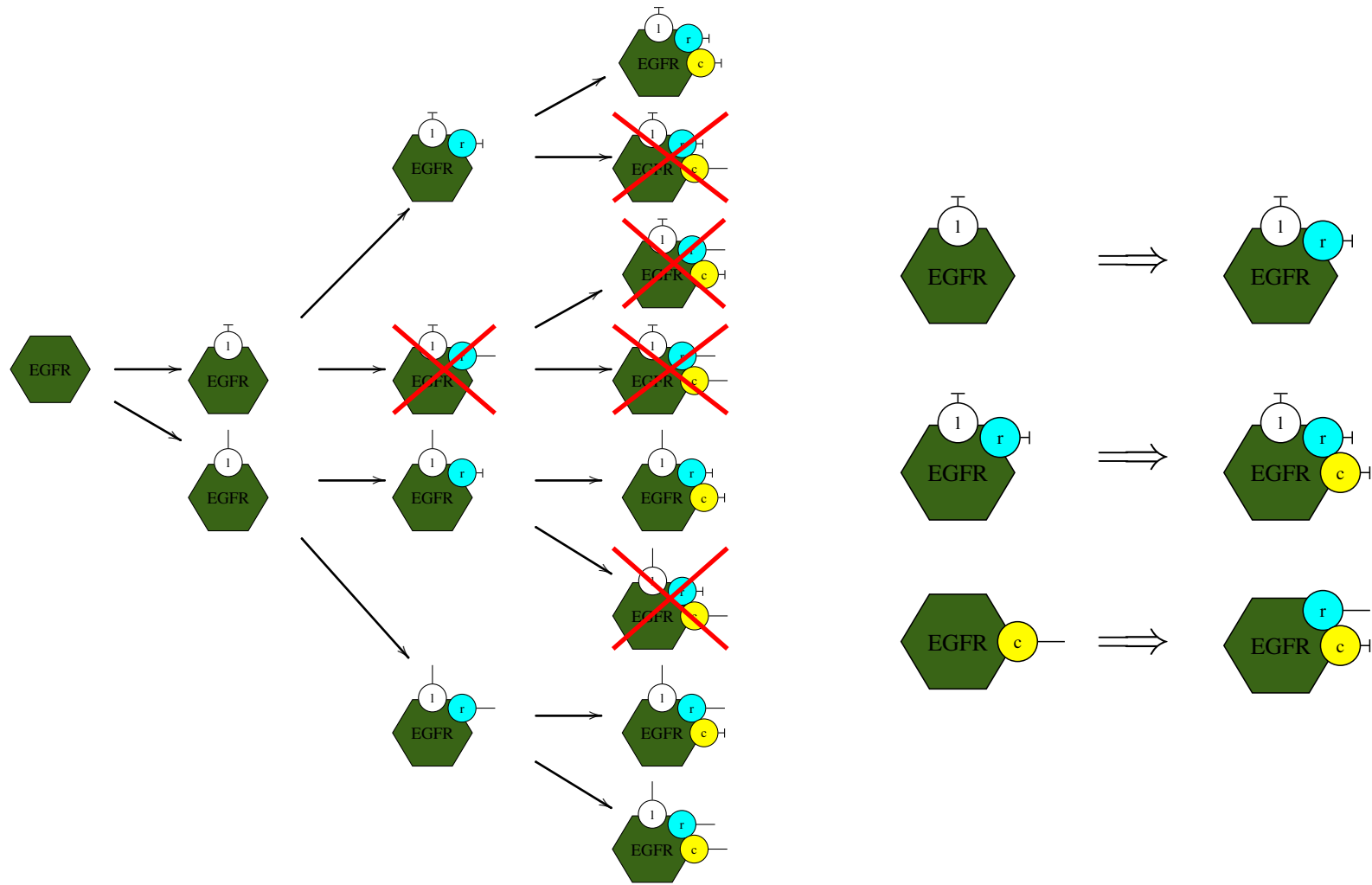
# Compatibility checking procedure



# Compatibility checking procedure



# Visualization



# Benchmarks

model	rules	inferred constraints	detected dead rules	analysis time (seconds)
repressilator	42	0	0	0.005
fceri_fyn_trimer	362	4	36	0.301
fceri_fyn_lyn_745	40	4	2	0.021
egfr	20	9	0	0.010
egfr, erk, mapk, ras	69	7	0	0.046
machine	220	13	7	0.405
ensemble	233	26	0	0.364
korkut (2017/01/17)	12896	0	874	24
korkut (2017/02/06)	5750	0	884	57
TGF (2018/04/19)	292	13	0	0.625
BigWnt (2017/03/22)	1486	14	12	8.74

on a MacBook Pro - Intel Core i7-6567U (3.3 GHz)

# Conclusion

- Mutual induction between several orthogonal sets;
- Automatic parameterization (by one-pass inspection of the rules);
- Accurate analysis (on the models we were given);
- Efficient analysis  
*but not enough to analyse big models in a text editor.*

# Perspectives

- Analysis of families of models;
- Incremental analysis.

# Overview

1. Introduction
2. Abstraction: Local views
3. Completeness: false positives?
4. Local fragment of Kappa
5. Other abstract domains
6. Decontextualization
7. Conclusion

# Outline

- we have a syntactic criterion in order to ensure that the set of reachable chemical species of a kappa system is local ;
- we now design program transformations to help systems satisfying this criterion ;
  1. **decontextualization**
    - is fully automatic;
    - preserves the transition system;
    - simplifies rules thanks to reachability analysis.
  2. **conjugation**
    - manual;
    - preserves the set of reachable chemical species;
    - uses backtrack to add new rules.



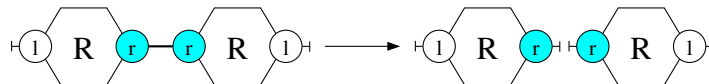
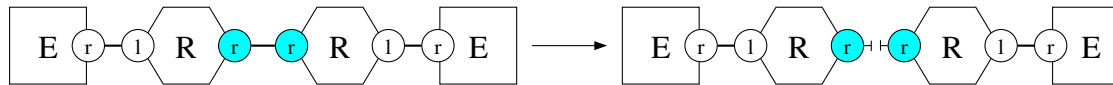
# Questions about sets of patterns

Many features that could help enhancing the confidence in models, such as:

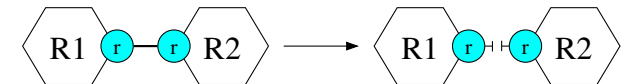
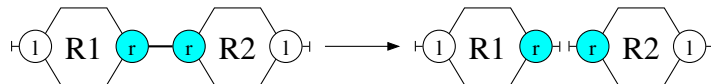
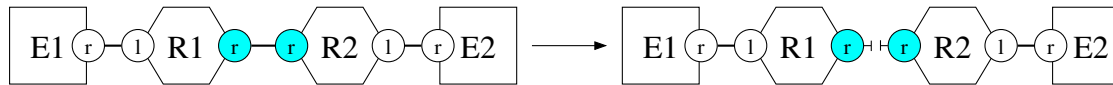
- knowing under which context a given interaction may not be applied?
- knowing under which common context a given interaction may be applied twice (by two distinct rules)?
- knowing under which context one application of a given rule may trigger the application of another given rule?

come down to the resolution of equations over sets of patterns.

# Are these two sets of rules equivalent?



# Identifying agents

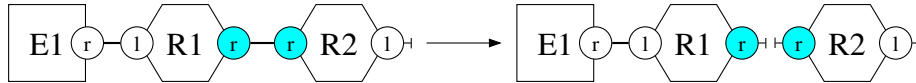


# Encoding in first order logic



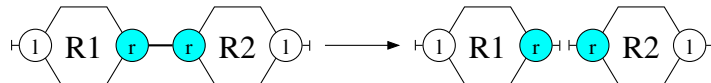
$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R \wedge \text{TYPE}(E1) = E \wedge \text{TYPE}(E2) = E$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = .\text{T.} \wedge \text{LINK}((E1, r), (R1, l)) = .\text{T.} \wedge \text{LINK}((E2, r), (R2, l)) = .\text{T.}$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R \wedge \text{TYPE}(E1) = E$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = .\text{T.} \wedge \text{LINK}((E1, r), (R1, l)) = .\text{T.} \wedge \text{STATE}(R2, l) = \neg$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R$$

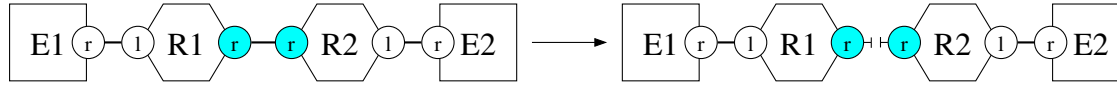
$$\wedge \text{LINK}((R1, r), (R2, r)) = .\text{T.} \wedge \text{STATE}(R1, l) = \neg \wedge \text{STATE}(R2, l) = \neg$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R$$

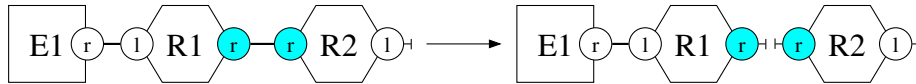
$$\wedge \text{LINK}((R1, r), (R2, r)) = .\text{T.}$$

# Reasonning up to iso



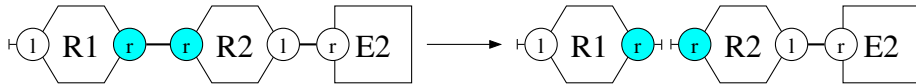
$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R \wedge \text{TYPE}(E1) = E \wedge \text{TYPE}(E2) = E$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = .T. \wedge \text{LINK}((E1, r), (R1, l)) = .T. \wedge \text{LINK}((E2, r), (R2, l)) = .T.$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R \wedge \text{TYPE}(E1) = E$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = .T. \wedge \text{LINK}((E1, r), (R1, l)) = .T. \wedge \text{STATE}(R2, l) = \neg$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R \wedge \text{TYPE}(E2) = E$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = .T. \wedge \text{LINK}((E2, r), (R2, l)) = .T. \wedge \text{STATE}(R1, l) = \neg$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R$$

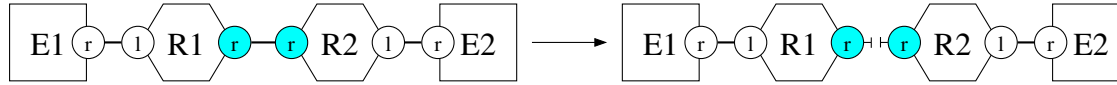
$$\wedge \text{LINK}((R1, r), (R2, r)) = .T. \wedge \text{STATE}(R1, l) = \neg \wedge \text{STATE}(R2, l) = \neg$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R$$

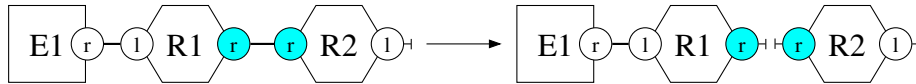
$$\wedge \text{LINK}((R1, r), (R2, r)) = .T.$$

# Using structural and model invariants



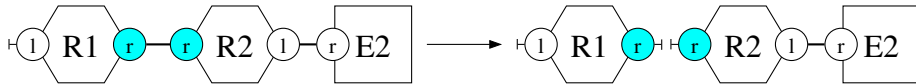
$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R \wedge \text{TYPE}(E1) = E \wedge \text{TYPE}(E2) = E$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = \text{.T.} \wedge \text{LINK}((E1, r), (R1, l)) = \text{.T.} \wedge \text{LINK}((E2, r), (R2, l)) = \text{.T.}$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R \wedge \text{TYPE}(E1) = E$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = \text{.T.} \wedge \text{LINK}((E1, r), (R1, l)) = \text{.T.} \wedge \text{STATE}(R2, l) = \neg$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R \wedge \text{TYPE}(E2) = E$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = \text{.T.} \wedge \text{LINK}((E2, r), (R2, l)) = \text{.T.} \wedge \text{STATE}(R1, l) = \neg$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = \text{.T.} \wedge \text{STATE}(R1, l) = \neg \wedge \text{STATE}(R2, l) = \neg$$



$$\text{TYPE}(R1) = R \wedge \text{TYPE}(R2) = R$$

$$\wedge \text{LINK}((R1, r), (R2, r)) = \text{.T.}$$

$$\wedge \rho_{\text{Kappa}} \wedge \rho_{\text{Model}}$$

# Issues

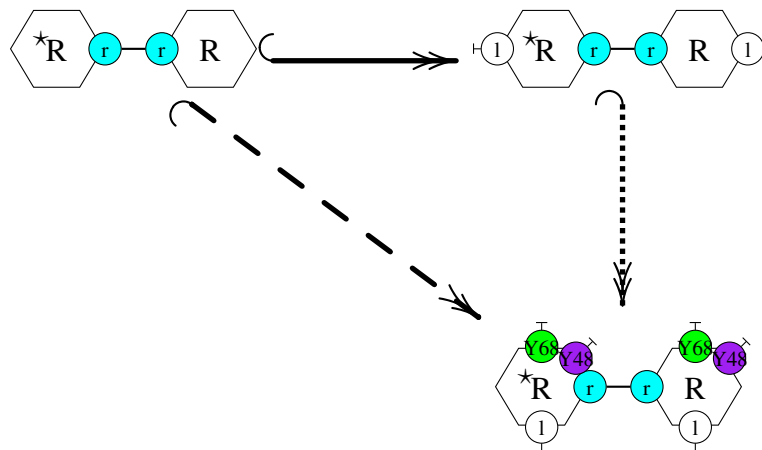
- How to align patterns?
- How to deal with agent identifiers?
- How to account with the constraints coming from Kappa and those coming from the models?

# Rooted pattern

- An **root** is a pattern of interest.
- An **anchorage** is an embedding from the root to a **rooted graph**.

An anchorage  $\mathcal{A}$  denotes the set  $[\mathcal{A}]$  of the embeddings from the root to fully specified patterns (mixtures) that can be factored by this anchorage.

For instance :



Such sets of embeddings may be combined by using set operations.



# Attributes about rooted mixtures

$$\text{TYPE}(id) = \begin{cases} A & \text{Agent } id \text{ is of type } A. \\ \perp & \text{There is no agent } id. \end{cases}$$

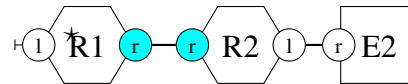
$$\text{STATE}(id, s) = \begin{cases} A'.s' & \text{The site } s \text{ of agent } id \text{ is bound to} \\ & \text{the site } s' \text{ of an agent of type } A'. \\ \neg & \text{The site } s \text{ of agent } id \text{ is free.} \\ \perp & \text{Either there is no agent } id, \\ & \text{or this agent has no site } s. \end{cases}$$

$$\text{LINK}((id, s), (id', s')) = \begin{cases} .T. & \text{The site } s \text{ of the agent } id \text{ is bound to} \\ & \text{the site } s' \text{ of the agent } id'. \\ .F. & \text{In any other case} \end{cases}$$

# Agent identifiers

- In the root, one agent per connected component is identified.
- Agent identifiers are relative paths starting from this agent.

For instance, in the following rooted pattern:



agent identifiers are defined as follows:

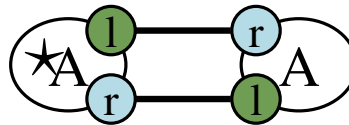
- $R1 := \varepsilon$ ;
- $R2 := \varepsilon.r - r$ ;
- $E2 := \varepsilon.r - r.l - r$ .

# Cyclic patterns

We stop every relative path after reaching an agent already in that path.

We use the attribute **ALIAS** to encode pairs of paths that denote the same agents.

For instance the following rooted pattern:



is encoded by the following formula:

$$\text{TYPE}(\varepsilon) = A \wedge \text{TYPE}(\varepsilon.r - l) = A \wedge \text{TYPE}(\varepsilon.l - r) = A \wedge \text{ALIAS}(\varepsilon, \varepsilon.r - l) = .F. \wedge \text{ALIAS}(\varepsilon.r - l, \varepsilon.l - r) = .T.$$

$$\text{ALIAS}(id, id') = \begin{cases} .T. & \text{Agents } id \text{ and } id' \text{ are the same} \\ .F. & \text{There is no agent } id, \text{ there is no agent } id', \\ & \text{or they are distinct.} \end{cases}$$

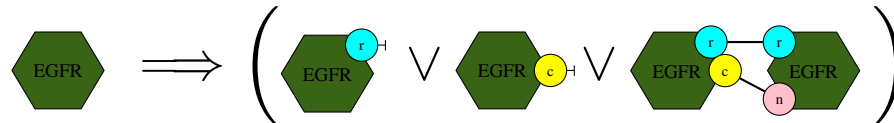
# Saturation lemmas

## 1. Structural lemmas:

- Relative paths also describe bonds;
- The prefix of a valid relative path is valid as well;
- Bonds are symmetric;
- Aliases propagate;
- Aliases propagate the other attributes;
- Information about bonds can be weakened;

## 2. Model invariants:

- Implications captured by static analysis, such as:



**Issue:** These lemmas are universally quantified over relative paths.

# A hierarchy of interpretations

- $\llbracket \phi \rrbracket$  is the set of the valuations that satisfy  $\phi$ .
- $\llbracket \phi \rrbracket_{\mathcal{K}}$  is the set of the valuations that satisfy  $\phi$  and that are the translation of a (rooted) mixture.
- $\llbracket \phi \rrbracket_{\mathcal{K}, \mathcal{M}}$  is the set of the valuations that satisfy  $\phi$  and that are the translation of a reachable (rooted) mixture.

$$\llbracket \phi \rrbracket = \llbracket \phi' \rrbracket \implies \llbracket \phi \rrbracket_{\mathcal{K}} = \llbracket \phi' \rrbracket_{\mathcal{K}}$$
$$\llbracket \phi \rrbracket_{\mathcal{K}} = \llbracket \phi' \rrbracket_{\mathcal{K}} \implies \llbracket \phi \rrbracket_{\mathcal{K}, \mathcal{M}} = \llbracket \phi' \rrbracket_{\mathcal{K}, \mathcal{M}}$$
Deciding whether  $\llbracket \phi \rrbracket_{\mathcal{K}, \mathcal{M}} = \llbracket \phi' \rrbracket_{\mathcal{K}, \mathcal{M}}$  is undecidable.

# Full abstraction

$$\llbracket \Psi_P \rrbracket_{\mathcal{K}, \mathcal{M}} = \llbracket \Psi_{P'} \rrbracket_{\mathcal{K}, \mathcal{M}} \iff [P] = [P']$$

where  $\Psi_P$  is the translation of the rooted pattern  $P$  in first order logic.

## Coherence

For any saturation lemma  $\rho$ ,

$$\llbracket \phi \rrbracket_{\mathcal{K}, \mathcal{M}} = \llbracket \phi \wedge \rho \rrbracket_{\mathcal{K}, \mathcal{M}}$$

Reminders:

- $[.]$  maps each rooted pattern to the underlying set of rooted reachable mixtures;
- $\llbracket . \rrbracket_{\mathcal{K}, \mathcal{M}}$  maps every formula to the set of the valuations that satisfy this formula and that are the translation of a reachable mixture.

# Boolean algebra

We define  $\mathcal{Id}(\phi)$  as the set of agent identifiers (access paths) that occur in  $\phi$ .

- $\uparrow(\phi) = (\rho_\phi, \phi \wedge \rho_\phi)$

where  $\rho_\phi = \bigwedge \{\rho \mid \rho \text{ saturation lemma s.t. } \mathcal{Id}(\rho) \subseteq \mathcal{Id}(\phi)\};$

- $\neg(\rho, \phi) = (\rho, (\neg \phi) \wedge \rho);$

- $(\rho_1, \phi_1) \wedge (\rho_2, \phi_2) = (\rho_1 \wedge \rho_2, (\phi_1 \wedge \rho_2) \wedge (\phi_2 \wedge \rho_1));$

- $(\rho_1, \phi_1) \vee (\rho_2, \phi_2) = (\rho_1 \wedge \rho_2, (\phi_1 \wedge \rho_2) \vee (\phi_2 \wedge \rho_1)).$

# Saturation: Soundness

For a given finite conjunction  $\rho$  of saturation lemmas.

$$\Phi(\psi_1 \wedge \rho, \dots, \psi_n \wedge \rho) \wedge \rho \equiv \Phi(\psi_1, \dots, \psi_n) \wedge \rho$$

Proof:

- $((\psi_1 \wedge \rho) \wedge (\psi_2 \wedge \rho)) \wedge \rho \equiv (((\psi_1 \wedge \psi_2) \wedge \rho) \wedge \rho) \wedge \rho$   
 $\equiv (\psi_1 \wedge \psi_2) \wedge \rho$
- $((\psi_1 \wedge \rho) \vee (\psi_2 \wedge \rho)) \wedge \rho \equiv ((\psi_1 \vee \psi_2) \wedge \rho) \wedge \rho$   
 $\equiv (\psi_1 \vee \psi_2) \wedge \rho$
- $(\neg(\psi_1 \wedge \rho)) \wedge \rho \equiv ((\neg \psi_1) \vee (\neg \rho)) \wedge \rho$   
 $\equiv ((\neg \psi_1) \wedge \rho) \vee ((\neg \rho) \wedge \rho)$   
 $\equiv ((\neg \psi_1) \wedge \rho)$



# Saturation: Parsimony

For a given finite conjunction  $\rho$  of saturation lemmas.

$$\Phi(\psi_1 \wedge \rho, \dots, \psi_n \wedge \rho) \equiv \Phi(\psi_1 \wedge \rho, \dots, \psi_n \wedge \rho) \wedge \rho$$

where  $\Phi$  contains no negation.

Proof:

- $((\psi_1 \wedge \rho) \wedge (\psi_2 \wedge \rho)) \equiv (\psi_1 \wedge \psi_2) \wedge \rho$
- $((\psi_1 \wedge \rho) \vee (\psi_2 \wedge \rho)) \equiv (\psi_1 \vee \psi_2) \wedge \rho$
- $\neg(\psi_1 \wedge \rho) \equiv (\neg \psi_1) \vee (\neg \rho)$

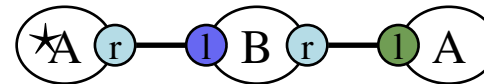
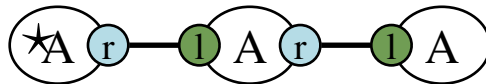
# Back to Kappa: conjunction of atoms (unfolded form)

$$\text{TYPE}(\varepsilon) = A \wedge \text{TYPE}(\varepsilon.l - r.l - r) = A \wedge \text{ALIAS}(\varepsilon, \varepsilon.l - r.l - r) = .T.$$

# Back to Kappa: conjunction of atoms (unfolded form)

$$\text{TYPE}(\varepsilon) = A \wedge \text{TYPE}(\varepsilon.l - r.l - r) = A \wedge \text{ALIAS}(\varepsilon, \varepsilon.l - r.l - r) = .T.$$

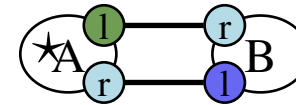
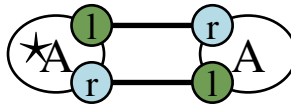
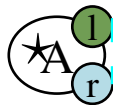
The type of the agent at the relative address  $\varepsilon.l - r$  is underspecified.  
We have to consider every solution according to the signature of the model:



# Back to Kappa: conjunction of atoms (Kappa patterns)

$$\text{TYPE}(\varepsilon) = A \wedge \text{TYPE}(\varepsilon.l - r.l - r) = A \wedge \text{ALIAS}(\varepsilon, \varepsilon.l - r.l - r) = .T.$$

Then, we fold agents according to the information about aliases.  
When underspecified, we consider every potential case.



# Back to Kappa: general case

Logical formulae are encoded into BDDs (MVBDUs).

## PROS:

- It offers canonic form (with respect to  $[[\cdot]]$ );
- Every formula is a disjunction of conjunctions;
- It offers a parsimonious representation (whenever an attribute is useless, it is discarded automatically);
- It uses optimal sharing and memoisation.

## CONS:

- Formulae (and their traduction back into Kappa) depend on an *a priori* order over the attributes;
- Worst case computation may be exponential in theory. Computation time is difficult to predict in practice.

# Conjugation

If a rule  $R'$  is equivalent to a rule in the transitive closure of the system.  
Then it may be included in the system without modifying reachable states.  
To remove the context  $C$  of a rule, we try to apply it for another context  $C'$  by:

1. removing the context  $C'$  (backtrack) ;
2. building the context  $C$  ;
3. applying the initial rule ;
4. removing the context  $C$  (backtrack) ;
5. building the context  $C'$ .

This is proved manually.

# Overview

1. Introduction
2. Abstraction: Local views
3. Completeness: false positives?
4. Local fragment of Kappa
5. Other abstract domains
6. Decontextualization
7. Conclusion

# Conclusion

- A scalable static analysis to abstract the reachable chemical species.
- A class of models for which the abstraction is complete.
- Many applications:
  - idiomatic description of reachable chemical species;
  - dead rule detection;
  - rule decontextualization;
  - computer-driven kinetic refinement.
- It can also help simulation algorithms:
  - wake up/inhibition map (agent-based simulation);
  - flat rule system generation (for bounded set of chemical species);
  - on the fly flat rule generation (for large/unbounded set)



# Bibliography

- Vincent Danos, Jérôme Feret, Walter Fontana, Jean Krivine: Abstract Interpretation of Cellular Signalling Networks. VMCAI 2008: 83-97
- Jérôme Feret, Kim Quyên Lý: Reachability Analysis via Orthogonal Sets of Patterns. SASB 2018: 27-48
- Pierre Boutillier, Ferdinanda Camporesi, Jean Coquet, Jérôme Feret, Kim Quyên Lý, Nathalie Théret, Pierre Vignat: KaSa: A Static Analyzer for Kappa. CMSB 2018: 285-291