Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Theoretical Computer Science 276 (2011) 29–59

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Formal Reduction for Rule-based Models

Ferdinanda Camporesi[1,2]

Dipartimento di Scienze dell'Informazione
Università di Bologna
Bologna, Italy
&
Laboratoire d'informatique de l'École normale supérieure
(INRIA/ÉNS/CNRS)
Paris, France

Jérôme Feret [1,3]

Laboratoire d'informatique de l'École normale supérieure
(INRIA/ÉNS/CNRS)
Paris, France

Abstract

Molecular biological models usually suffer from a large combinatorial explosion. Indeed, proteins form complexes and modify each others, which leads to the formation of a huge number of distinct chemical species (i.e. non-isomorphic connected components of proteins). Thus we cannot generate explicitly the quantitative semantics of these models, and even less compute their properties.
In this paper we propose a formal framework to automatically reduce the combinatorial complexity of the differential semantics of rule-based models. Our reduction is based on two abstractions, which are combined thanks to a generic product. The first abstraction tracks the flow of information between the different regions of chemical species, so as to detect and abstract away some useless correlations between the state of sites. The second abstraction detects pairs of sites having the same capabilities of interaction, and abstracts away any distinction between them. The initial semantics and the reduce one are formally related by Abstract Interpretation.

Keywords: rules-based modeling, model reduction, abstract interpretation, flow of information.

1 Introduction

Modelers of molecular signaling networks must cope with the combinatorial explosion of protein states generated by post-translational modifications and complex

[1] The contribution of Ferdinanda Camporesi and Jérôme Feret was partially supported by the AbstractCell ANR-Chair of Excellence.

[2] Email: campores@di.ens.fr

[3] Email: feret@di.ens.fr

doi:10.1016/j.entcs.2011.09.014

formations. Rule-based models provide a powerful alternative to approaches that require an explicit enumeration of all possible chemical species of a system [6,1]. Such models consist of formal rules stipulating the (partial) contexts for specific protein-protein interactions to occur. The behavior of the models can be formally described by stochastic or differential semantics. Yet, the naive computation of these semantics does not scale to large systems, because it does not exploit the lower resolution at which rules specify interactions.

We present a formal framework for constructing coarse-grained differential semantics. We instantiate this framework with two abstract domains. The first one tracks information flow between the different regions of chemical species, so as to detect and abstract away some useless correlations between the state of sites. The second one detects pairs of sites having the same capabilities of interaction and abstracts away any distinction between them.

The result of our abstraction is a set of chemical patterns, called fragments, and a system which describes exactly the concentration evolution of these fragments. The method never requires the execution of the concrete rule-based model and the soundness of the approach is described and proved by abstract interpretation [4].

Related work.

In [3] is proposed a framework where the information flow between the sites of chemical species is used so as to build reduced models. With this approach there is no formal definition for the semantics or for the flow of information. Moreover, reduced models have to be written by hand.

In [7], a framework is proposed to automatically derive reduced models from sets of rules. The semantics of reduced models are formally related to the semantics of the unreduced ones. The framework that we present here is an extension of this framework. Unlike in [7], our fragments are heterogeneous. The cutting of a protein into portions may depend on its position within the chemical species. This matches more closely with the flow of information. Indeed, within a chemical species, the behavior of a protein may be driven by the state of a site without being driven by the state of the same site in other instances of the protein. Our new analysis exploits this efficiently. In [10], another family of fragments are defined, with an even higher level of context-sensitivity. The set of fragments is computed iteratively by building overlaps between connected components in rules and already built fragments. It is not clear whether this approach scales to large models, or not. In our approach, we have taken an appropriate trade-off of context-sensitivity: we first compute very fastly an over-approximation of the flow of information, from which we deduce an efficient symbolic description of the set of fragments. In [14], a language independent approach is described. Yet it requires an extensional description of rules as sets of reactions, and fragments as multi-set of species, which makes the approach impractical for large systems. Lastly, in [9,8], fragments are use to reduce the dimension of the stochastic semantics of rule-based models.

Outline.

In the Section 2, we describe some case study to illustrate our approach. In the Section 3, we provide a generic framework to define differential semantics, reduce these semantics, and combine these reductions. In the Section 4, we introduce the language Kappa and its differential semantics. In the Section 5, we show how to detect pairs of sites having the same capabilities of interaction and we use this information to design a model reduction. In the Section 6, we introduce an analysis of the flow of information between the different regions of chemical species, and deduce which correlations can be abstract away. Then, we use this information to cut chemical species into self-consistent fragments.

2 Case study

Let us start out with some motivating examples.

2.1 Symmetric sites

This first example illustrates that we can detect when some sites have the same capabilities of interaction, and use this to abstract away any distinction between these sites.

We consider four kinds of chemical species: P, *P, P*, and *P*. These are four instantiations of a given protein P which bears two activation sites. Each site can be activated (which is denoted by the symbol '$\star$' on the left or on the right according to which site is activated), or not. Initially, all proteins have no activated site. The evolution of the state of the proteins is described thanks to some chemical reactions. There is no order in the activation of the sites. A first site (either the left or the right one) can be activated at rate k_1 thanks to the reactions in the Figure 1(a) (the rates specify the speed of the reactions). Then the other site can be activated at rate k_2 thanks to the reactions in the Figure 1(b). Once both sites are activated the protein can be destroyed at rate k_3 by the reaction in the Figure 1(c).

The differential semantics of this model is the solution of the system of ordinary differential equations (ODEs) which is given in the Figure 1(d). This system is obtained by applying Mass Action Law. It describes the continuous evolution of the concentration of each chemical species along the time. Intuitively, Mass Action Law states that the amount of time a reaction is applied within a small amount of time is obtained by multiplying the rate constant of the reaction by the product of the concentration of the reactants (which are the chemical species which occur in the left hand side of the reaction).

We notice that, in a protein, both sites have the same capabilities of interaction. Thus we propose to ignore any distinction between these two sites. Indeed, what is important is not which sites are activated in a given protein, but how many sites are activated. Doing this, we get the system of equations in the Figure 1(e). This system can be derived analytically from the system given in the Figure 1(d). We observe a reduction of the dimension of the state space. In a more general setting, if the protein had n such sites, there would be 2^n chemical species, but only $(n+1)$

$$\mathrm{P} \longrightarrow {}^{\star}\mathrm{P} \qquad k_1$$
$$\mathrm{P} \longrightarrow \mathrm{P}^{\star} \qquad k_1$$

(a) First activation.

$${}^{\star}\mathrm{P} \longrightarrow {}^{\star}\mathrm{P}^{\star} \qquad k_2$$
$$\mathrm{P}^{\star} \longrightarrow {}^{\star}\mathrm{P}^{\star} \qquad k_2$$

(b) Second activation.

$${}^{\star}\mathrm{P}^{\star} \longrightarrow \qquad k_3$$

(c) Destruction.

$$\begin{cases} [\mathrm{P}]' = -2k_1[\mathrm{P}] \\ [{}^{\star}\mathrm{P}]' = k_1[\mathrm{P}] - k_2[{}^{\star}\mathrm{P}] \\ [\mathrm{P}^{\star}]' = k_1[\mathrm{P}] - k_2[\mathrm{P}^{\star}] \\ [{}^{\star}\mathrm{P}^{\star}]' = k_2([{}^{\star}\mathrm{P}] + [\mathrm{P}^{\star}]) - k_3[{}^{\star}\mathrm{P}^{\star}] \end{cases}$$

(d) Initial differential system.

$$\begin{cases} [\mathrm{P}]' = -2k_1[\mathrm{P}] \\ [{}^{\star}\mathrm{P} + \mathrm{P}^{\star}]' = k_1[\mathrm{P}] - k_2([{}^{\star}\mathrm{P}] + [\mathrm{P}^{\star}]) \\ [{}^{\star}\mathrm{P}^{\star}]' = k_2([{}^{\star}\mathrm{P}] + [\mathrm{P}^{\star}]) - k_3[{}^{\star}\mathrm{P}^{\star}] \end{cases}$$

(e) Reduced differential system.

Fig. 1. Chemical reactions and ODEs for the protein with two symmetric sites.

variables in the reduced system.

We have seen through this example how the fact that several sites may have the same capabilities of interaction allows the inference of a changement of variables which reduces the model.

2.2 Hierarchic flow of information

Now we consider an example where a changement of variables can be deduced from an over-approximation of the flow of information among sites.

We consider a protein having three activation sites r, c, l, each of which can be activated 'p', or deactivated 'u'. Thus a chemical species is denoted as a triple of symbols among 'u' and 'p', the first component denotes the state of the site l, the second one the state of the site c, and the third one the state of the site r. Initially, all proteins have no activated site. The evolution of the state of the proteins is described thanks to some chemical reactions. There is some hierarchic control between the states of the sites. The site c has to be activated first, at rate k_1, thanks to the reaction in the Figure 2(a). Once the site c has been activated, the l site can get activated at rate k_2, no matter the state of the site r is (see the Figure 2(b)); and the site r can get activated at rate k_3, no matter the state of the site l is (see the Figure 2(c)). We describe the flow of information among the states of the sites of a protein in the Figure 2(d). Intuitively, the flow of information summarizes the fact that the state of the site c may control the behavior of the states of the sites l and r, but that the states of the sites l and r do not control the behavior of the states of the other sites.

The differential semantics of this model is the solution of the system of ODEs which is given in the Figure 2(e), where the concentration of the protein in the state (x_1, x_2, x_3) is denoted by $[x_1, x_2, x_3]$. Since the state of the site l does not control the evolution of the state of the site r, and conversely, we can abstract away the correlation between the states of the sites l and r. To do this, we cut the chemical species into fragments, each fragment documenting either the sites l and c, or the sites c and r. Such a fragmentation defines a linear changement of variables.

$$(\mathrm{u,u,u}) \longrightarrow (\mathrm{u,p,u}) \quad k_1$$

(a) 2nd site activation.

$$(\mathrm{u,p,u}) \longrightarrow (\mathrm{p,p,u}) \quad k_2$$

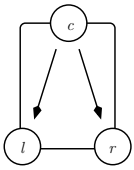
$$(\mathrm{u,p,p}) \longrightarrow (\mathrm{p,p,p}) \quad k_2$$

(b) 1st site activation.

$$(\mathrm{u,p,u}) \longrightarrow (\mathrm{u,p,p}) \quad k_3$$

$$(\mathrm{p,p,u}) \longrightarrow (\mathrm{p,p,p}) \quad k_3$$

(c) 3rd site activation.



(d) Flow of information.

$$\begin{cases} [u,u,u]' = -k_1[u,u,u] \\ [u,p,u]' = -k_2[u,p,u] + k_1[u,u,u] - k_3[u,p,u] \\ [u,p,p]' = -k_2[u,p,p] + k_3[u,p,u] \\ [p,p,u]' = k_2[u,p,u] - k_3[p,p,u] \\ [p,p,p]' = k_2[u,p,p] + k_3[p,p,u] \end{cases}$$

(e) Initial differential system.

$$\begin{cases} [u,u,u]' = -k_1[u,u,u] \\ ([u,p,u] + [u,p,p])' = k_1[u,u,u] - k_2([u,p,u] + [u,p,p]) \\ ([p,p,u] + [p,p,p])' = k_2([u,p,u] + [u,p,p]) \\ ([u,p,u] + [p,p,u])' = k_1[u,u,u] - k_3([u,p,u] + [p,p,u]) \\ ([u,p,p] + [p,p,p])' = k_3([u,p,u] + [p,p,u]) \end{cases}$$

(f) Reduced differential system.

Fig. 2. Chemical reactions, flow of information, and ODEs for the protein with hierarchic flow of information

Indeed we can define the concentration of a fragment, as the linear combination of the concentration of the chemical species in which this fragment occurs. For instance, the concentration of the fragment which documents the sites l and c, and where both these sites are activated is equal to the sum of the concentrations of the chemical species (p,p,u) and (p,p,p). Applying this changement of variables, we get the reduced system which is given in the Figure 2(f). We notice that the number of variables in the two systems are the same, because of the simplicity of the example. In practice, abstracting away a correlation reduces a lot the number of variables.

We have seen in this example, that an over-approximation of the flow of information between the sites of chemical species, can be used to identify useless correlations, which can be used to discover appropriate changement of variables.

2.3 Dimers

Our third example illustrates the weakness of our previous approach [7,5] to reduce models where the states of some sites flow across binding between proteins.

We consider a kind of receptors, which when activated, can form dimers and initiate other cascades of interactions. More precisely, we consider a protein having four interaction sites a, b, c, d. The sites a, c, and d can be activated ('p') or not ('u'), while the site b is a binding site: the sites b of two receptors can be bound together. We call a dimer the gathering of two receptors. With these constraints, we can form exactly 38 chemical species (either single receptors, or dimers, in various configurations according to the states of the sites a, c, and d). Thus, describing

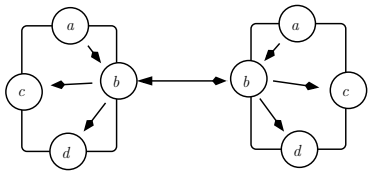


Fig. 3. Flow of information within the sites of a dimer.

explicitly the reactions of the system would be cumbersome. So we describe these reactions only implicitly: the site a of any receptor can get activated at rate k_1; two receptors having their site a activated can bind to each other at rate k_2; then, when both sites a of a dimer are still activated, the site c can be activated at rate k_3 and the site d at rate k_4; lastly, all these reactions are reversible: any activated site can become deactivated at rate k_5 for the site a, k_7 for the site c, and k_8 for the site d, and dimers may break their binding at rate k_6.

We notice that the behavior of the site c (resp. d) does depend neither on the state of the site d (resp. c) of the same receptor, nor on the state of the site c and d of the potential partner in case of dimer. But, in a dimer, the state of the site a of a receptor controls the evolution of the sites c and d of the other receptor. We say, that there is a flow of information across bindings. An over-approximation of the flow of information between the sites of dimers is given in the Figure 2.3. We can use the framework in [7,5], to cut down the combinatorial complexity of this example. Indeed, in this framework, we use a set of fragments, the evolution of the concentration of which can be defined in a self-consistent way. These fragments are homogeneous, because the way proteins are cut into portions of proteins is defined once for all: in our case, each portion of protein will document either the states of the sites a, b, and c; or the states of the sites a, b, and d. As a consequence, the fragments of dimer will all document 6 sites. Thus the fact that the states of the sites c or d of a given receptor cannot control the behavior of the sites c and d in the other receptor of a dimer is not exploited, which is a severe limitation of the framework in [7,5]. In the framework that we are presenting in this paper, the way proteins are cut is not the same for all the proteins of a chemical species. We are using heterogeneous fragments: in the fragments of dimer, one receptor is privileged and documents three sites (a, b, and either c and d), while the other documents only the sites a and b. This cutting of chemical species into fragments matches more closely with the flow of information.

We have seen in this example that the framework that is proposed in [7,5], can be improved by using heterogeneous fragments, where the cut of proteins is not the same throughout a chemical species.

3 Model reduction of differential semantics

3.1 Notations

Let A be a set. We define a state over A as a mapping between A and $\mathbb{R}$, such that we have $\rho(A) \geqslant 0$ for any element $a \in A$. Let B be another set. We consider two

norms $||\cdot||_A$ and $||\cdot||_B$ respectively over A and B. Let ϕ be a mapping between $A \to \mathbb{R}$ and $B \to \mathbb{R}$. The mapping ϕ is called an abstraction between $A \to \mathbb{R}$ and $B \to \mathbb{R}$ if and only the following properties are satisfied: (i) ϕ is a linear mapping between $A \to \mathbb{R}$ and $B \to \mathbb{R}$; (ii) for any state ρ over A, the element $\phi(\rho)$ is a state over B; and (iii) for any sequence $(\rho_n)_{n\in\mathbb{N}}$ of states such that the sequence $(||\rho_n||_A)_{n\in\mathbb{N}}$ diverges, then the sequence $(||\phi(\rho_n)||_B)_{n\in\mathbb{N}}$ diverges as well. In such a case, we write:

$$(A \to \mathbb{R}) \overset{\phi}{\multimap} (B \to \mathbb{R}).$$

We notice that whenever the sets A and B are both finite, then the property (iii) does not depend on the choice of the norms $||\cdot||_A$ and $||\cdot||_B$.

3.2 Concrete semantics

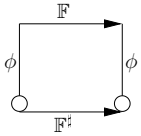
We define an autonomous system as a pair $(\mathcal{V}, \mathbb{F})$ where $\mathcal{V}$ is a finite set of variables and $\mathbb{F}$ is a continuously differentiable function from $\mathcal{V} \to \mathbb{R}$ to $\mathcal{V} \to \mathbb{R}$. By the Cauchy-Lipschitz theorem[11], for any state ρ_0 over $\mathcal{V}$, the system of equations:

$$\begin{cases} \rho'(t) = \mathbb{F}(\rho(t)) \\ \rho(0) = \rho_0 \end{cases}$$

has a unique maximal differentiable solution: $f_{\rho_0} : [0, T_{\rho_0}) \to (\mathcal{V} \to \mathbb{R})$, with $T_{\rho_0} \leqslant +\infty$. An autonomous system $(\mathcal{V}, \mathbb{F})$ is said to be positive, if and only if, for any state ρ_0 over $\mathcal{V}$ and any $t \in [0, T_{\rho_0})$, $f_{\rho_0}(t)$ is a state over $\mathcal{V}$ as well. The *concrete semantics* of the system $(\mathcal{V}, \mathbb{F})$ is the mapping $[\![\mathcal{V}, \mathbb{F}]\!]$ which associates the unique maximal differentiable solution f_{ρ_0} to each state ρ_0 over $\mathcal{V}$ (in this context, ρ_0 is called the initial state).

3.3 Exact reduction of differential semantics

A model reduction $(\mathcal{V}, \mathbb{F}, \mathcal{V}^\sharp, \phi, \mathbb{F}^\sharp)$ is a tuple such that the pair $(\mathcal{V}, \mathbb{F})$ is an autonomous system, $\mathcal{V}^\sharp$ is a finite set of variables, ϕ is an abstraction function between $\mathcal{V} \to \mathbb{R}$ and $\mathcal{V}^\sharp \to \mathbb{R}$, and $\mathbb{F}^\sharp$ a continuously differentiable function from $\mathcal{V}^\sharp \to \mathbb{R}$ to $\mathcal{V}^\sharp \to \mathbb{R}$, and such that the following square commutes:



The trajectories in the semantics of the (abstract) autonomous system $(\mathcal{V}^\sharp, \mathbb{F}^\sharp)$ are the exact projections by ϕ of the trajectories in the semantics of the (concrete) autonomous system $(\mathcal{V}, \mathbb{F})$, as stated by the following theorem which is proved in [5].

Theorem 3.1 *Let ρ_0 be an initial state over $\mathcal{V}$. We introduce T_{ρ_0} an $T^\sharp_{\phi(\rho_0)}$ such that $[0, T_{\rho_0})$ is the definition domain of $[\![\mathcal{V}, \mathbb{F}]\!](\rho_0)$ and $[0, T^\sharp_{\phi(\rho_0)})$ is the definition domain of $[\![\mathcal{V}^\sharp, \mathbb{F}^\sharp]\!](\phi(\rho_0))$.*

Then, under these assumptions, $T_{\rho_0} = T^\sharp_{\phi(\rho_0)}$ *and, for any* $t \in [0, T_{\rho_0})$, $\phi([\![\mathcal{V}, \mathbb{F}]\!](\rho_0)(t)) = [\![\mathcal{V}^\sharp, \mathbb{F}^\sharp]\!](\phi(\rho_0))(t)$.

If follows from the Theorem 3.1 that if the system $(\mathcal{V}, \mathbb{F})$ is positive, then the system $(\mathcal{V}^\sharp, \mathbb{F}^\sharp)$ is positive as well.

3.4 Projections-based reductions

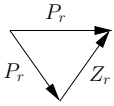
Now we investigate a specific class of model reductions: we focus on the case when an equivalence relation over the variables of the autonomous system can be lifted to a bisimulation, and use this to define a model reduction.

We consider a concrete autonomous system $(\mathcal{V}, \mathbb{F})$. We consider r a function from $\mathcal{V}$ to $\mathcal{V}$ such that r is idempotent (i.e. $r \circ r = r$). The function r defines an equivalence relation $\sim_r$ over $\mathcal{V}$ by $v_1 \sim_r v_2$ if and only if $r(v_1) = r(v_2)$. Moreover, for any variable $v \in \mathcal{V}$, the variable $r(v)$ is called the representative of the equivalence class of v. We define two linear projections P_r and Z_r over $\mathcal{V} \to \mathbb{R}$ as follows:

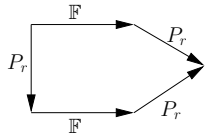
$$P_r(\rho) : \begin{cases} \mathcal{V} \to \mathbb{R} \\ v \mapsto \sum\{\rho(v') \mid r(v') = v\} \end{cases} \qquad Z_r(\rho) : \begin{cases} \mathcal{V} \to \mathbb{R} & \\ v \mapsto \rho(v) & \text{if } r(v') = v \\ v \mapsto 0 & \text{otherwise.} \end{cases}$$

Intuitively, P_r gathers the values of each $\sim_r$-equivalent variable, and stores the result to the value of the representative of each $\sim_r$-equivalence class, while Z_r ignores the values of the variables which are not the representative of their $\sim_r$-equivalence class.

We notice that $(\mathcal{V} \to \mathbb{R}) \xrightarrow{P_r}\!\!\!\!\circ\ (\mathcal{V} \to \mathbb{R})$ and that the following diagram commutes:

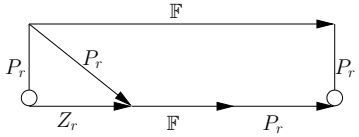


We says that the relation r induces a bisimulation over the autonomous system $(\mathcal{V}, \mathbb{F})$ if and only if for any pair (ρ, ρ') of states over $\mathcal{V}$, if $P_r(\rho) = P_r(\rho')$, then $P_r(\mathbb{F}(\rho)) = P_r(\mathbb{F}(\rho'))$. Equivalently, the relation r induces a bisimulation, if and only if the following diagram commutes:



Theorem 3.2 *Whenever* r *induces a bisimulation, then* $(\mathcal{V}, \mathbb{F}, \mathcal{V}, P_r, P_r \circ \mathbb{F} \circ Z_r)$ *is a model reduction.*

Proof. The proof is given by the following commutative diagram:



□

Example 3.3 Let us consider the example of the Section 2.1. The set of variables $\mathcal{V}$ is defined as $\{[\mathrm{P}], [^{\star}\mathrm{P}], [\mathrm{P}^{\star}], [^{\star}\mathrm{P}^{\star}]\}$. We want to identify the proteins which have only one site activated, no matter which site it is. Thus, we define the mapping r by $r([\mathrm{P}]) = [\mathrm{P}]$, $r([^{\star}\mathrm{P}]) = r([\mathrm{P}^{\star}]) = [^{\star}\mathrm{P}]$, and $r([^{\star}\mathrm{P}^{\star}]) = [^{\star}\mathrm{P}^{\star}]$. By using a matrix notation, we get:

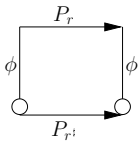
$$\mathbb{F} = \begin{bmatrix} -2k_1 & 0 & 0 & 0 \\ k_1 & -k_2 & 0 & 0 \\ k_1 & 0 & -k_2 & 0 \\ 0 & k_2 & k_2 & -k_3 \end{bmatrix}, \mathbb{F}^{\sharp} = \begin{bmatrix} -2k_1 & 0 & 0 & 0 \\ 2k_1 & -k_2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & k_2 & 0 & -k_3 \end{bmatrix}, P_r = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \text{ and } Z_r = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and we can check that $\mathbb{F}^{\sharp} = P_r \times \mathbb{F} \times Z_r$.

3.5 Combining a model reduction with projections-based reductions

An existing model reduction can be abstracted further thanks to a bisimulation induced by an equivalence relation over the concrete variables.

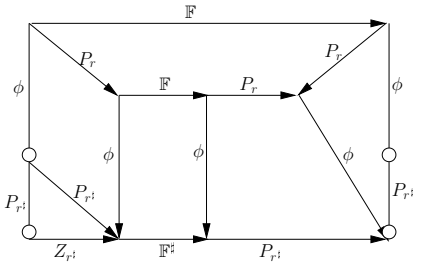
Theorem 3.4 *Let $(\mathcal{V}, \mathbb{F}, \mathcal{V}^{\sharp}, \phi, \mathbb{F}^{\sharp})$ be a model reduction, r be an idempotent mapping r over $\mathcal{V}$ such that r induces a bisimulation over the autonomous system $(\mathcal{V}, \mathbb{F})$, and $r^{\sharp}$ be an idempotent mapping over $\mathcal{V}^{\sharp}$. We assume that the following square:*



commutes.

Under these assumptions, the tuple $(\mathcal{V}, \mathbb{F}, \mathcal{V}^{\sharp}, P_{r^\sharp} \circ \phi, P_{r^\sharp} \circ \mathbb{F}^{\sharp} \circ Z_{r^\sharp})$ is a model reduction.

Proof. The proof is given by the following commutative diagram:



□

Interestingly, we notice that no commutative diagram was required to relate the functions $\mathbb{F}^{\sharp}$ and $P_{r^\sharp}$. Thus, we only need to prove that r induces a bisimulation in the concrete. Then, to inherit this construction we need to prove that $P_{r^\sharp} \circ \phi = \phi \circ P_r$. Such a proof is quite easy, since only the structure of the abstract variables matters, and not their dynamics.

$$
\begin{array}{rll}
a & ::= & N_l(\sigma) \qquad \text{(agent)} \\
N & ::= & A \in \mathcal{A} \qquad \text{(agent type)} \\
l & ::= & i \in \mathbb{N} \mid \overline{i} \in \overline{\mathbb{N}} \qquad \text{(agent identifier)} \\
\sigma & ::= & \varepsilon \mid s,\sigma \qquad \text{(interface)} \\
s & ::= & n_\iota^\lambda \qquad \text{(site)} \\
n & ::= & x \in \mathcal{S} \qquad \text{(site name)} \\
\lambda & ::= & \epsilon \mid N_l@n \mid N@n \mid - \mid ? \qquad \text{(binding state)} \\
\iota & ::= & \epsilon \mid \mathtt{w} \in \mathbb{I} \qquad \text{(internal state)}
\end{array}
$$

Fig. 4. Syntax for agents.

4 The Kappa language

Now, we instantiate the generic framework that we have proposed in the Section 3 with a particular language. We focus our study to the models that are written in Kappa [6]. In the present section, we present Kappa and its semantics.

Kappa is a graph-rewriting-based language. It has a graphical notation that eases the design of models. Nevertheless, we use here a process-algebra notation where agents are identified, which facilitates the presentation of the semantics and the various analyses.

4.1 Syntax

We fix a finite set of agent types $\mathcal{A}$, a finite set of sites $\mathcal{S}$, and a finite set $\mathbb{I}$ of non empty strings. We also consider two signature maps Σ_ι and Σ_λ assigning a set of sites to each agent type such that for any agent type $A \in \mathcal{A}$. Intuitively, $\Sigma_\iota(A)$ is the set of sites which can bear a modifiable internal state $w \in \mathbb{I}$ (such as a level of energy), whereas $\Sigma_\lambda(A)$ is the set of sites which can be bound to some other sites. We also denote by Σ the signature map that associates to each agent type $A \in \mathcal{A}$ the combined interface $\Sigma_\iota(A) \cup \Sigma_\lambda(A)$. The syntax of agents is given in the Figure 4.

An *agent identifier* l belongs to the set $\mathbb{N}$ of natural numbers, or to a copy $\overline{\mathbb{N}}$ of the set of natural numbers. Most agents will be identified by natural numbers. Identifiers in $\overline{\mathbb{N}}$ will be used temporary when agents are created, before a proper identifier is allocated.

An *interface* σ is a sequence of sites with internal states (as subscript) and binding states (as superscript). The internal state of the site s may be written as s_ϵ, which means that either it does not have internal states (when $s \in \Sigma(A) \backslash \Sigma_\iota(A)$), or it is not specified. A site that bears an internal state $\mathtt{w} \in \mathbb{I}$ is written $s_\mathtt{w}$ (in such a case $s \in \Sigma_\iota(A)$). A site can be free, or bound (which is possible only when $s \in \Sigma_\lambda(A)$). There are also several levels of information about binding states. We use a *question mark* '?' if we do not know anything about the binding state; we

use the symbol 'ϵ', if we know that the site is *free*. There are also several levels of information about bound sites: we use a *site address* $A_l@x$ if we know the binding partner (this means that the site is bound to the site x of the agent A with identifier l); we use a *binding type* $A@x$ if we only know that the partner is some site x of some agent A; lastly we use a *wildcard bond* '$-$' if we only know that a site is bound but have no further information about its partner. We generally omit the symbol 'ϵ' in examples.

An *agent* is given by a type A in $\mathcal{A}$, an agent identifier l and an interface σ. Such an agent is denoted by $A_l(\sigma)$. An *expression* E is a sequence of agents such that (i) no two agents have both the same type and the same identifier; (ii) no site name occurs more than once in a given interface; (iii) each site name s occurring in the interface of the agent A occurs in $\Sigma(A)$; (iv) each site name s which occurs in the interface of the agent A with an internal state distinct from 'ϵ' occurs in $\Sigma_\iota(A)$; (v) each site name which occurs in the interface of the agent A with a binding state distinct from 'ϵ' occurs in $\Sigma_\lambda(A)$. Furthermore, given an expression E and an agent type $A \in \mathcal{A}$, we denote by $\mathbf{agents}(E, A)$ the set of identifiers l such that there is an agent A in the expression E with identifier l.

A *pattern* is an expression E such that whenever the binding state of the site x in the agent of type A with identifier l is $A'_{l'}@x'$, then there exists an agent of type A' with the identifier l' such that this agent has the site x' in its interface and that the binding state of this site is $A_l@x$ (thus site addresses encode a pairing relation between some sites). A *proper pattern* is a pattern where each agent is identified with a proper identifier (in $\mathbb{N}$). A *mixture* E is a proper pattern that is fully specified, that is to say that each agent of type A in a mixture E documents its full interface $\Sigma(A)$, sites can only be free or bear a site address, and any sites in $\Sigma_\iota(A)$ have a non empty internal state. A pattern E is said to be *disconnected* if there is a strict subsequence E' of it that is a non-empty pattern. A *pattern component* is a connected pattern. A *species* is a non-empty connected mixture.

A rule is given by a pair of patterns (E_ℓ, E_r) and a rate k (which is a non negative real number), that is written $E_\ell \xrightarrow{k} E_r$, with some additional constraints explained below. The left hand side (lhs) E_ℓ of a rule describes the agents taking part in it and various conditions on both their internal and binding states for the rule to apply. The right hand side (rhs) describes what the rule does.

Definition 4.1 In a rule $E_\ell \xrightarrow{k} E_r$, firstly agents in the lhs are identified with natural numbers $i \in \mathbb{N}$ and secondly the pattern E_r is obtained from E_ℓ in the following stepwise fashion (the order matters):
- *(i) creation*: some agents $A_{\bar{i}}(\sigma)$ with an agent identifier in $\overline{\mathbb{N}}$, with their full interfaces $\Sigma(A)$, with all sites free and with all sites $s \in \Sigma_\iota(A)$ having a non empty internal state are added;
- *(ii) unbinding*: some occurrences of the wildcard '$-$' and some site addresses $A_i@n$ are removed;
- *(iii) deletion*: some agents with only free sites are removed;
- *(iv) modification*: some (non empty) internal states are replaced with (non empty)

$$P_1(l_u,r_u) \xrightarrow{k_1} P_1(l_p,r_u) \qquad P_1(l_u,r_u) \xrightarrow{k_1} P(l_u,r_p)$$

(a) First activation.

$$P_1(l_p,r_u) \xrightarrow{k_2} P_1(l_p,r_p) \qquad P_1(l_u,r_p) \xrightarrow{k_2} P_1(l_p,r_p)$$

(b) Second activation.

$$P_1(l_p,r_p) \xrightarrow{k_3} \varepsilon$$

(c) Destruction.

Fig. 5. Encoding of the example of the Section 2.1 in Kappa. The signature is defined as: $\mathcal{A} = \{P\}$, $\mathbb{I} = \{u,p\}$, $\mathcal{S} = \{l,r\}$, $\Sigma_\lambda(P) = \varnothing$, $\Sigma_\iota(P) = \{l,r\}$.

$$P_1(c_u) \xrightarrow{k_1} P_1(c_p)$$

(a) 2nd site activation.

$$P_1(l_u,c_p) \xrightarrow{k_2} P_1(l_p,c_p)$$

(b) 1st site activation.

$$P_1(c_p,r_u) \xrightarrow{k_3} P_1(c_p,r_p)$$

(c) 3rd site activation.

Fig. 6. Encoding of the example of the Section 2.2 in Kappa. The signature is defined as: $\mathcal{A} = \{P\}$, $\mathbb{I} = \{u,p\}$, $\mathcal{S} = \{l,c,r\}$, $\Sigma_\lambda(P) = \varnothing$, $\Sigma_\iota(P) = \{l,c,r\}$.

internal states;
- *(v) binding*: some free sites are bound pair-wise by using appropriate site addresses.

Agent types and identifiers ensure a 1-1 mapping correspondence between the agents in the lhs and in the rhs that are neither removed, nor created. Moreover, this correspondence preserves the set of sites which are documented in interfaces, and the set of sites which carry a non empty internal state.

Note that according to the Definition 4.1, only the bonds that are denoted by a pair of site addresses, or a wildcard '$-$' can be released, unlike binding types which can only be tested.

Example 4.2 Let us encode in Kappa the three examples of the Section 2. The example with the two symmetric sites is written in the Figure 5. We notice that in this example, we do not take benefit of the context freeness of Kappa: indeed, there are as many Kappa rules as there are chemical reaction and the Kappa rules document exactly the same amount of information as the chemical reactions. The example with the hierarchic flow of information is given in the Figure 6. In this example, there are fewer rules than reactions and they do not document all the sites of the protein. For instance, in the Figure 6(b), we use the fact that the state of the third site does not control the activation of the first site, so as to gather the two reactions in the Figure 2(b) into a single one which does not document the state of the third site. The example of the dimer is given in the Figure 7. This last example illustrates well the fact that Kappa allows for a compact description of models. By using context-free rule, we have described a model which involves 38 chemical species. The rule in the Figure 7(a) stipulates that the site a of a receptor can get activated. Then two activated receptors may bind to each other thanks to the rule in the Figure 7(b). When the two receptors in a dimer are still activated, then one receptor in the dimer may activate the site c or d in the other receptor, as stated by the rules in the Figure 7(c). At any moment, activated sites can get deactivated and dimers may break their connexion as stated in the Figure 7(d).

$$R_1(a_u) \xrightarrow{k_1} R_1(a_p)$$
(a) Receptor activation.

$$R_1(a_p,b)\ ,\ R_2(a_p,b) \xrightarrow{k_2} R_1(a_p,b^{A_2@b})\ ,\ R_2(a_p,b^{A_1@b})$$
(b) Dimerisation.

$$R_1(a_p,b^{A_2@b},c^u)\ ,\ R_2(a_p,b^{A_1@b}) \xrightarrow{k_3} R_1(a_p,b^{A_2@b},c^p)\ ,\ R_2(a_p,b^{A_1@b})$$
$$R_1(a_p,b^{A_2@b},d^u)\ ,\ R_2(a_p,b^{A_1@b}) \xrightarrow{k_4} R_1(a_p,b^{A_2@b},d^p)\ ,\ R_2(a_p,b^{A_1@b})$$
(c) Cross-activation.

$$R_1(a_p) \xrightarrow{k_5} R_1(a_u) \qquad R_1(b^-) \xrightarrow{k_6} R_1(b)$$
$$R_1(c_p) \xrightarrow{k_7} R_1(c_u) \qquad R_1(d_p) \xrightarrow{k_8} R_1(d_u)$$
(d) Relaxation.

Fig. 7. Encoding of the example of the Section 2.3. The signature is defined as: $\mathcal{A} = \{R\}$, $\mathbb{I} = \{u,p\}$, $\mathcal{S} = \{a,b,c,d\}$, $\Sigma_\lambda(R) = \{b\}$, $\Sigma_\iota(R) = \{a,c,d\}$.

$$E, N_l(\sigma,s,s',\sigma'), E' \equiv E, N_l(\sigma,s',s,\sigma'), E'$$
$$E, a, a', E' \equiv E, a', a, E'$$
(a) Structural congruence.

$$\lambda = \epsilon, N@n, -, ? \implies \overline{\phi}(\lambda) = \lambda$$
$$\overline{\phi}(N_l@n) = N_{\phi(N,l)}@n$$
$$\overline{\phi}(n_\iota^\lambda) = n_\iota^{\overline{\phi}(\lambda)}$$
$$\overline{\phi}(s,\sigma) = \overline{\phi}(s),\overline{\phi}(\sigma)$$
$$\overline{\phi}(N_l(\sigma)) = N_{\phi(N,l)}\big(\overline{\phi}(\sigma)\big)$$
(b) Agent substitution.

$$\iota_\ell = \iota, \epsilon \implies \iota \models \iota_\ell$$
$$\lambda = \lambda_\ell \implies \lambda \models \lambda_\ell$$
$$N_l@n \models N@n$$
$$N_l@n \models -$$
$$\lambda \models ?$$
$$\iota \models \iota_\ell \wedge \lambda \models \lambda_\ell \models n_\iota^\lambda \models n_{\iota_\ell}^{\lambda_\ell}$$
$$\sigma \models \varepsilon$$
$$s \models s_\ell \wedge \sigma \models \sigma_\ell \implies s,\sigma \models s_\ell,\sigma_\ell$$
$$\sigma \models \sigma_\ell \implies N_l(\sigma) \models N_l(\sigma_\ell)$$
(c) Agent matching.

$$\iota[\epsilon] = \iota \qquad \iota[\mathtt{W}_r] = \mathtt{W}_r$$
$$\lambda[\epsilon] = \epsilon \qquad \lambda[N_l@n] = N_l@n$$
$$\lambda[N_l@n] = \lambda$$
$$\lambda[-] = \lambda$$
$$\lambda[?] = \lambda$$
$$n_\iota^\lambda[n_{\iota_r}^{\lambda_r}] = n_{\iota[\iota_r]}^{\lambda[\lambda_r]}$$
$$\sigma[\varepsilon] = \sigma$$
$$(s,\sigma)[s_r,\sigma_r] = s[s_r],\sigma[\sigma_r]$$
$$N_l(\sigma)[N_l(\sigma_r)] = N_l(\sigma[\sigma_r])$$
(d) Agent replacement.

Fig. 8. Structural congruence, substitution, matching and replacement. Definitions are made by induction over the syntax.

4.2 Operational semantics

Now we define the operational semantics of sets of rules.

First, we define the application of a rule $E_\ell \xrightarrow{k} E_r$ to a proper pattern E.

Informally, one needs to embed E_ℓ into E. For that purpose, we define a *substitution* as a partial mapping ϕ between pairs $(A,l) \in \mathcal{A} \times (\mathbb{N} \cup \overline{\mathbb{N}})$ of agent type/identifier and agent identifiers $l' \in \mathbb{N} \cup \overline{\mathbb{N}}$. A substitution ϕ is clean when for any couple $(A,\bar{l}) \in (\mathcal{A} \times \overline{\mathbb{N}}) \cap \mathbf{dom}(\phi)$, $\phi(A,\bar{l}) = \bar{l}$. A substitution ϕ can be applied with a pattern E if, and only if, for any agent type $A \in \mathcal{A}$, we have $(A,l) \in \mathbf{dom}(\phi)$ for any agent identifier $l \in \mathbf{agents}(E,A)$. Indeed applying a substitution ϕ consists in replacing the agent identifier l with the agent identifier $\phi(A,l)$, in the agent of type A and identifier l (if it exists). This is formalized, in the Figure 8(b), by defining the extension $\overline{\phi}$ of ϕ to agents. Furthermore, a given substitution ϕ is *into* if, and only if, for any agent type A, and any two identifiers l,l', we have $\phi(A,l) = \phi(A,l') \implies l = l'$. An into substitution ϕ is a candidate for identifying the agents of two patterns. More precisely, each agent $A_{l_\ell}(\sigma_\ell)$ in the first pattern can be identified with the agent $A_l(\sigma)$, if (i) agent identifiers are the same (ie $l = \phi(A,l_\ell)$) and (ii) the signature σ contains more information than the signature $\overline{\phi}(\sigma_\ell)$. The second property is formalized by a matching relation $\models$ which is given in the Figure 8(c). Yet, since interfaces are defined up to permutations of sites, one may have to reorder the sites before applying the matching relation, thanks to the congruence relation $\equiv$ which is defined in the Figure 8(a).

We can now properly define an embedding between two patterns. An embedding ϕ between two patterns E_ℓ and E is an into substitution such that: (i) $\mathbf{dom}(\phi) = \{(A,l) \mid A \in \mathcal{A}, l \in \mathbf{agents}(E_\ell,A)\}$, (ii) and for any $(A,l) \in \mathbf{dom}(\phi)$, there exists an agent a' such that $a \equiv a'$ and $a' \models \overline{\phi}(a_\ell)$, where a_ℓ is the unique agent in E_ℓ of type A with identifier l and a the unique agent in E of type A with identifier $\phi(A,l)$. A clean embedding is a clean into substitution. Moreover, whenever there exist an embedding ϕ between E and E_ℓ, and an embedding ϕ' between E_ℓ and E, we say that ϕ is an *isomorphic embedding*. We notice that any embedding between two species is an isomorphic embedding. Given a pattern E, we define the number of symmetries in E as the number of embeddings ϕ such as E and $\phi(E)$ are $\equiv$-equivalent. We denote the number of symmetries of E as $\mathbf{sym}(E)$.

Now we define the impact of applying the rule $E_\ell \xrightarrow{k} E_r$ along a given clean embedding ϕ between the lhs E_ℓ of the rule and a pattern E. For that purpose we consider three kinds of agents:
- Agents $A_{i_\ell}(\sigma_\ell)$ are said to be preserved if, and only if, $A_{l_\ell}(\sigma_\ell)$ occurs in E_ℓ and there exists an interface σ_r such that $A_{l_\ell}(\sigma_r)$ occurs in E_r;
- Agents $A_{l_r}(\sigma_r)$ are said to be created if, and only if, $A_{l_r}(\sigma_r)$ occurs in E_r, but there is no agent of type A with identifier l_r in E_ℓ;
- Agents $A_{l_\ell}(\sigma_\ell)$ are said to be removed if, and only if, $A_{l_\ell}(\sigma_\ell)$ occurs in E_ℓ, but there is no agent of type A with identifier l_ℓ in E_r.

We extend the clean embedding ϕ so as to deal with newly created agents. Thus, we define the clean into substitution $\phi^\star$ over $\mathbf{dom}(\phi) \cup (\mathcal{A} \times \overline{\mathbb{N}})$ by $\phi^\star(A,i) = \phi(A,i)$ for any $(A,i) \in \mathbf{dom}(\phi)$ and $\phi^\star(A,\bar{i}) = \bar{i}$ otherwise (this way, $\phi^\star$ preserves temporary identifiers). Then, for any agent $A_{l_\ell}(\sigma_\ell)$ that is removed, the agent of type A with identifier $\phi(A,l_\ell)$ is removed in E; for any agent a_r that is created, the agent $\overline{\phi^\star}(a_r)$ is added in E; last for any preserved agent a_ℓ, we denote by a_r and a the agents in

E_r and E which have the same type and the same identifier as the agent a_ℓ, then we select an agent a' (the choice does not matter) such that $a \equiv a'$ and $a' \models \overline{\phi}(a_\ell)$, then the agent a is replaced with agent $a'[\overline{\phi^\star}(a_r)]$, where .[.] is a replacement function that is defined in the Figure 8(d). We denote by $E[E_r]_\phi$ the so obtained expression (which is well defined up to $\equiv$-equivalence).

One shall notice that $E[E_r]_\phi$ might be not a pattern, because there might be some pending bonds which are sites with a binding state of the form $A_l@x$ but, either the agent of type A and identifier l has been removed, or the site x of the agent of type A and identifier l has been made free. Thus, we remove pending bonds: we introduce the function **clean** between patterns such that $\mathbf{clean}(E)$ is obtained by replacing with the symbol ϵ, each site address $A_l@x$ such that either there is no agent of type A and identifier l in E, or the site x of the agent of type A with identifier l is free.

In the case when the proper pattern E is a mixture, we expect the result of the application of a rule to be a mixture as well. Yet, we notice that $\mathbf{clean}(E[E_r]_\phi)$ might be not a mixture because of temporary identifiers. We have to allocate fresh proper identifiers for the newly created agents: we introduce the function **fresh** between patterns, such that $\mathbf{fresh}(E)$ is obtained by replacing any temporary agent identifier $\bar{i}$ of the agent A by $M(A) + i + 1$ where $M(A)$ is the maximum element of the non empty finite set $\{0\} \cup (\mathbb{N} \cap \mathbf{agents}(E, A))$.

Now we can define the operational semantics as a labeled transition system. The states of the system are mixtures (up to $\equiv$). We shall notice that the impact of applying a rule $E_\ell \xrightarrow{k} E_r$ on a mixture E is fully defined (up to $\equiv$) by the clean embedding ϕ between the lhs E_ℓ of the rule and the mixture E. So we define the set $\mathcal{L}$ of labels as the set of the tuples (r, E, ϕ) where r is a rule $E_\ell \xrightarrow{k} E_r$, E is a state, ϕ is an embedding between E_ℓ and E. In such a case, we write:

$$E \xrightarrow{(r,E,\phi)} \mathbf{fresh}(\mathbf{clean}(E[E_r]_\phi)).$$

Example 4.3 Let us consider the following example. We set $\mathcal{A} = \{A\}$, $\mathbb{I} = \varnothing$, $\mathcal{S} = \{a, b\}$, $\Sigma_\lambda(A) = \{a, b\}$, $\Sigma_\iota(A) = \varnothing$. We consider the mixture $E = A_1\left(a^{A_2@\mathrm{b}}, b^{A_3@\mathrm{a}}\right)$, $A_2\left(a, b^{A_1@\mathrm{a}}\right)$, $A_3\left(a^{A_1@\mathrm{b}}, b\right)$ and the rule $A_1(a) \xrightarrow{k} A_{\overline{0}}(a, b)$. Intuitively, this rule can be applied with an agent of type A, the site a of which is free (whatever the state of the site b is). Moreover, this rule removes the agent $A_1(a)$, and replace it with a new agent $A_{\overline{0}}(a, b)$ of type A with both site a and b free (in this rule, no agent is preserved).

There exists only one embedding between the lhs $A_1(a)$ of the rule and the mixture E. Namely, $\phi = [A, 1 \mapsto 2]$ (neither the substitution $[A, 1 \mapsto 1]$, nor $[A, 1 \mapsto 3]$ is an embedding, since the site a is free neither in the agent $A_1\left(a^{A_2@b}, b^{A_3@a}\right)$, nor in the agent $A_3\left(a^{A_1@b}, b\right)$). Moreover, the expression $E[E_\ell]_\phi$ is equal to the expression $A_1\left(a^{A_2@b}, b^{A_3@a}\right)$, $A_3\left(a^{A_1@b}, b\right)$, $A_{\overline{0}}(a, b)$. This expression has a pending bond (on the site a of agent A_1), which is removed by the primitive **clean**. Indeed the expression $\mathbf{clean}(E[E_r]_\phi)$ is equal to the expression $A_1\left(a, b^{A_3@a}\right)$, $A_3\left(a^{A_1@b}, b\right)$, $A_{\overline{0}}(a, b)$. Then the temporary identifier $\overline{0}$ is replaced

with 4 by the primitive **fresh**. As the result, the expression $\mathbf{fresh}(\mathbf{clean}(E[E_r]_\phi))$ is equal to the expression $A_1\big(a,b^{A_3@a}\big)$, $A_3\big(a^{A_1@b},b\big)$, $A_4(a,b)$.

A rule can be more or less refined. There are basically two ways of refining a rule, either we add more information on the lhs of the rule (which is called a left refinement), and report it on the rhs, or we add more information on the rhs and report it to the lhs (which is called a right refinement).

More formally, let $r := E_\ell \xrightarrow{k} E_r$ be a rule. Given an embedding ϕ between E_ℓ and a pattern E, we define the left refinement of r via the embedding ϕ as the rule $E \xrightarrow{k'} \mathbf{clean}(E[E_r]_\phi)$, where k' is equal to the product between k and the ratio between the number of symmetries in E and E_ℓ. Moreover given a clean embedding ϕ' between E_r and a pattern E', there exists a pattern E and an embedding ϕ between E_ℓ and E such that $\phi(A,i) = \phi'(A,i)$ for any $(A,i) \in \mathbf{dom}(\phi')$ such that $i \in \mathbb{N}$, we define the right refinement of r via the embedding ϕ' as the left refinement of r via the embedding ϕ.

4.3 Differential semantics

Now we remind the differential semantics of Kappa [7,5].

We consider a set of rules $\mathcal{R}$ and a finite set of species $\mathcal{V}$, which is closed under the rules in $\mathcal{R}$ and has at most one representative per species isomorphism class. More formally, (i) for any mixture E, any pattern component of which is isomorphic to an element in $\mathcal{V}$, any rule $E_\ell \xrightarrow{k} E_r$, and any embedding ϕ between E_ℓ and E, each pattern component in $\mathbf{clean}(E[E_r]_\phi)$ is isomorphic to an element in $\mathcal{V}$; and (ii) for any pair (v,v') of elements in $\mathcal{V}$, if there exists an embedding between v and v', then $v = v'$.

The states ρ of the system are mapping between chemical species $v \in \mathcal{V}$ and real numbers in $\mathbb{R}$. ($\rho(v)$ denotes the concentration of the species v). So as to define the function $\mathbb{F}$ which specifies the behavior of the system, we consider the set of chemical reactions which are generated by the set of rules $\mathcal{R}$. Given a rule $r := E_\ell \xrightarrow{k} E_r$ in $\mathcal{R}$, we may assume without any loss of generality that E_ℓ is written as $C_1,\dots,C_k$ where each C_i is a pattern component. A reaction is obtained, by choosing for any integer i between 1 and k, a reachable species $v_i \in \mathcal{V}$ and an embedding ϕ_i between C_i and v_i. The expression $v_1,\dots,v_k$ might not be a mixture because distinct species may share some agent identifiers. In order to define the product of a reaction, we choose k species $w_1,\dots,w_k$ and k embedding $\psi_1,\dots,\psi_k$ such that $w_1,\dots,w_k$ is a mixture, and that for any i between 1 and k, ψ_i is an embedding between v_i and w_i. This way, we form a composite embedding $\phi = \sum_i \psi_i \circ \phi_i$ between E_ℓ and $w_1,\dots,w_k$. The result of the application of the rule r on $w_1,\dots,w_k$ along ϕ is isomorphic to a tuple of species in $\mathcal{V}$ that we denote by $p_1,\dots,p_l$ (we can check that $p_1,\dots,p_l$ does not depend on the choice of the $w_1,\dots,w_k$).

Then the function $\mathbb{F}$ is obtained by summing the contribution of each reaction,

as follows:

$$\mathbb{F}(\rho)(v_j) \stackrel{-}{=} \gamma \prod_i (\rho(v_i) \mid 1 \leqslant i \leqslant k), \qquad \mathbb{F}(\rho)(p_{j'}) \stackrel{+}{=} \gamma \prod_i (\rho(v_i) \mid 1 \leqslant i \leqslant k).$$

where γ is the quotient between k and the number of symmetries in E_ℓ and j ranges between 1 and k, and j' between 1 and l.

The obtained autonomous system $(\mathcal{V}, \mathbb{F})$ is positive. Indeed, for any species $v \in \mathcal{V}$, $\mathbb{F}(v)$ can be written as $-\rho(v) \cdot P[\rho(v_1), \ldots, \rho(v_m)] + Q[\rho(v'_1), \ldots, \rho(v'_n)]$, where P and Q are two polynomial mapping with positive coefficients and $v_1, \ldots, v_m, v'_1, \ldots, v'_n$ are $m + n$ species in $\mathcal{V}$.

5 Symmetric sites in Kappa

5.1 Action of a transposition

In this section, we formalize the actions of a transposition of two sites on patterns and rules. Then we define when two sites are symmetric in a given set of rules.

We consider two kinds of transformation of pattern. The first one, called *transposition of binding types* consists in replacing a site name with another one in an instance of a binding type and the second one, called *transposition of states* consists in permuting the states of two sites in one agent.

More precisely, a *transposition of binding types* is defined as a tuple $(A, l, z, B, x, y) \in \mathcal{A} \times \mathbb{N} \times \mathcal{S} \times \mathcal{A} \times \mathcal{S} \times \mathcal{S}$, such that $z \in \Sigma_\lambda(A)$ and $x, y \in \Sigma_\lambda(B)$. A *transposition of states* is defined as a tuple $(A, l, x, y) \in \mathcal{A} \times (\mathbb{N} \cup \overline{\mathbb{N}}) \times \mathcal{S} \times \mathcal{S}$, such that the following properties are satisfied: (i) the site x belongs to the set $\Sigma(A)$; (ii) the site x belongs to $\Sigma_\iota(A)$ if and only if the site y belongs to $\Sigma_\iota(A)$; (iii) the site x belongs to $\Sigma_\lambda(A)$ if and only if the site y belongs to the set $\Sigma_\lambda(A)$. A transposition is either a transposition of binding types, or a transposition of states. The set of all transpositions is denoted by $\mathbb{T}$.

Now we describe the action of transposition on patterns. A transposition of binding type $t := (A, l, z, B, x, y)$ operates on a pattern E in the following way: if E contains an agent A with identifier l documenting the site z, then if the binding state of z is the binding type $B@x$, then it is replaced with the binding type $B@y$, else if the binding state of z is the binding type $B@y$, then it is replaced with the binding type $B@x$. In any other cases, E is not modified.

The transposition of sites (A, l, x, y) denotes that we want to permute the internal state and the binding state of the sites x and y in the agent A with identifier l, if such an agent occurs in a pattern. Thus, whenever there is no agent A with identifier l in the pattern E then the pattern E remains unchanged. Otherwise, the transformation is defined in two steps. First we define E' as the expression which is obtained by replacing any instance of a site address $A_l@x$ with the site address $A_l@y$ and *vice versa*. Let us write E' as a sequence $a'_1, \ldots, a'_n$ of agents. We know that there exists a unique agent a'_k in E' of type A and identifier l. Let us write $a'_k = A_l(\sigma)$. We define the expression E'' by replacing in E' the agent a'_k with the agent $A_l(\sigma')$ where σ' is defined as the interface where $\sigma'(x)$ is defined and equal to

$\sigma(y)$ if and only if $\sigma(y)$ is defined; $\sigma'(y)$ is defined and equal to $\sigma(x)$ if and only if $\sigma(x)$ is defined, and for any site in $\Sigma(A)\backslash\{x,y\}$, $\sigma'(z)$ is defined and equal to $\sigma(z)$ if and only if $\sigma(z)$ is defined. The expression E'' is then called the result of the application of the transposition t on the pattern E.

Given a transposition, we denote by $\mathbf{subs}(\mathbf{t},\mathbf{E})$ the result of the application of the transposition t on the pattern E. We notice that $\mathbf{subs}(\mathbf{t},\mathbf{E})$ is a pattern. Moreover, if E is a proper pattern (resp. a mixture, resp. a species), then $\mathbf{subs}(\mathbf{t},\mathbf{E})$ is a proper pattern (resp. a mixture, resp. a species) as well.

Example 5.1 We consider the following signature: $\mathcal{A} = \{A,B\}$, $\mathcal{S} = \{x,y,z\}$, $\mathbb{I} = \varnothing$, $\Sigma_\iota(A) = \Sigma_\iota(B) = \varnothing$, and $\Sigma_\lambda(A) = \Sigma_\lambda(B) = \{x,y,z\}$.

We consider the pattern $E := A_1\left(x^{A_2@y}\right)$, $A_2\left(y^{A_1@x},z^{B@x}\right)$. Then, applying the transposition of binding types $t_1 := (A,2,z,B,x,y)$ to E replaces the binding type $B@x$ with the binding type $B@y$ in the agent A with identifier 2: $\mathbf{subs}(\mathbf{t_1},\mathbf{E}) = A_1\left(x^{A_2@y}\right)$, $A_2\left(y^{A_1@x},z^{B@y}\right)$. Moreover, applying the transposition of states $t_2 := (A,1,x,y)$ to E is computed in two steps. Firstly we replace the site address $A_1@x$ with the site address $A_1@y$, secondly we replace the site name x with the site name y in the agent A with identifier 1. Thus, we get: $\mathbf{subs}(\mathbf{t_2},\mathbf{E}) = A_1\left(y^{A_2@y}\right)$, $A_2\left(y^{A_1@y},z^{B@x}\right)$. Lastly, applying the transposition of states $t_3 := (A,2,y,z)$ is computed in two steps. Firstly we replace the site address $A_2@y$ with the site address $A_2@z$, secondly we swap the states of the site y and of the site z in the agent A with identifier 2. Thus, we get: $\mathbf{subs}(\mathbf{t_3},\mathbf{E}) = A_1\left(x^{A_2@z}\right)$, $A_2\left(z^{A_1@x},y^{B@x}\right)$.

Now we consider a rule $r := E_\ell \xrightarrow{k} E_r$ and a well-defined transposition t. The rule:

$$r' := \mathbf{subs}(\mathbf{t},\mathbf{E}_\ell) \xrightarrow{k'} \mathbf{subs}(\mathbf{t},\mathbf{E_r}), \text{where } k' = k \cdot \frac{\mathbf{sym}(\mathbf{subs}(\mathbf{t},\mathbf{E}_\ell))}{\mathbf{sym}(E_\ell)},$$

is well-defined. In such a case, the rule r' is called the *action of the transposition t on the rule r*, and is denoted by $\mathbf{subs}_{\mathcal{R}}(\mathbf{t},\mathbf{r_1})$.

5.2 Definition of symmetric sites

We use transpositions in order to identify the sites having the same capabilities of interaction. The idea is the following: let us fix an agent type A and two sites x and y in $\Sigma(A)$ such that $x \in \Sigma_\lambda(A)$ if and only if $y \in \Sigma_\lambda(A)$, and $x \in \Sigma_\iota(A)$ if and only if $y \in \Sigma_\iota(A)$. So as to detect whether x and y have the same capabilities of interaction, we will replace each rule with the combination of rules which can be obtained in substituting zero, one, or several occurrences of x with y, and zero, one, or several occurrences of y with x. If the obtained system of rules is equivalent to the initial one, then the sites x and y have the same capabilities of interaction. Special care has to be taken about the kinetic rates of rules. When a rule is replaced with n rules (up to transposition of x and y in the instances of A), then the rate of each rule has to be divided by n. Moreover, in order to show that the initial and the obtained

systems are equivalent, one may have to reorder interfaces and reindex agents in rules (by applying a same into substitution to both sides of a given rule) and gather some rules having the same lhs and the same rhs (summing up the rates).

More precisely, given a rule r and a non negative real number $k \in \mathbb{R}^+$, we define $\mathbf{scale}(r,k)$ as the rule that is obtained by multiplying the rate of r by k. Moreover, we define the orbit of the rule r, as the set, which is written $\mathbf{orbit}(r)$, of rules which can be obtained by applying zero, one, or several transpositions of states to the rule r. Since the lhs and the rhs of a rule are finite expressions, the orbit of a rule is always a finite set. Our model transformation is formalized by the binary relation $\Longrightarrow$ over sets of rules, which is defined as follows:

$$\mathcal{R} \Longrightarrow \left\{ \mathbf{scale}(r', \frac{1}{\mathbf{card}(\mathbf{orbit}(r))}) \;\middle|\; r \in \mathcal{R},\ r' \in \mathbf{orbit}(r) \right\}.$$

for any set of rules $\mathcal{R}$.

Agents and sites in rules can be reordered using the congruence relation over their both hand sides. Moreover, agents can be reindexed using substitutions. We use a slight extension of the substitution that we have used in the Section 4.1, since we may need to reindex temporary identifiers (in $\overline{\mathbb{N}}$). A generalized substitution is a mapping ϕ between $\mathbb{N} \cup \overline{\mathbb{N}}$ and $\mathbb{N} \cup \overline{\mathbb{N}}$, such that for any proper identifier $l \in \mathbb{N}$, $\phi(l) \in \mathbb{N}$ and such that for any temporary identifier $\bar{l} \in \overline{\mathbb{N}}$, $\phi(\bar{l}) \in \overline{\mathbb{N}}$. A generalized substitution is into if and only if for any identifier $l, l' \in \mathbb{N} \cup \overline{\mathbb{N}}$, $\phi(l) = \phi(l') \Longrightarrow l = l'$. The extension $\overline{\phi}$ of a generalized substitution ϕ to agents is defined as in the case of substitution (eg. see the Section 4.1). We define an equivalence relation $\approx$ over rules such that two rules $r := E_\ell \xrightarrow{k} E_r$ and $r' := E'_\ell \xrightarrow{k'} E'_r$ are $\approx$-equivalent whenever $k = k'$ and there exists an into generalized substitution ϕ such that $\overline{\phi}(E_\ell) \equiv E'_\ell$ and $\overline{\phi}(E_r) \equiv E_r$.

Two set of rules are equivalent whenever they can be made equal by replacing their rules with $\approx$-equivalent ones and by gathering the rules having the same lhs and the same rhs (in such a case, their rates are summed up).

Example 5.2 We consider two examples of rule sets. The first one is the example of the Section 2.1, in which we do not assume that the rates of the first two reactions are the same. The second example is a more subtle example.

The examples and their automatic transformation are given in the Figure 9. For each, the initial set of rules is given on the left, and the transformed one (after reordering and reindexing) is given on the right. In the Figure 9(a), we notice that whenever the rates k_1 and k_2 are equal then the left and the right systems are equal. Thus under this assumption, the sites x and y are symmetric in A. Let us detail the transformation, the first two rules are replaced with two rules each, with a half rate, one applying on the site x, and the other applying on the site y. The four obtained rules are pairwisely equal, so we gather them pairwise. When transforming the third rule, we obtain two equivalent rules (with half rate), that we can gather to recover the initial rule.

In the Figure 9(b), we notice that the sites x and y are symmetric whenever

$$\begin{array}{lcl} A_1(x_u) \xrightarrow{k_1} A_1(x_p) & & A_1(x_u) \xrightarrow{k'} A_1(x_p) \\ A_1(y_u) \xrightarrow{k_2} A_1(y_p) & \Longrightarrow & A_1(y_u) \xrightarrow{k'} A_1(y_p) \\ A_1(x_p,y_p) \xrightarrow{k_3} \varepsilon & & A_1(x_p,y_p) \xrightarrow{k_3} \varepsilon \end{array}$$

$$\text{where } k' = \frac{k_1}{2} + \frac{k_2}{2}.$$

(a) In this first example, the signature is: $\mathcal{A} = A$, $\mathcal{S} = \{x, y\}$, $\mathbb{I} = \{u, p\}$, $\Sigma_\iota(A) = \{x, y\}$, and $\Sigma_\lambda(A) = \varnothing$. The set of rules on the left is transformed into a set of rules which is equivalent to the one on the right. We can conclude that the sites x and y are symmetric in A whenever $k_1 = k_2$.

$$\begin{array}{lcl} A_1(x)\ ,\ A_2(x) \xrightarrow{k_1} A_1\big(x^{A_2@x}\big)\ ,\ A_2\big(x^{A_1@x}\big) & & A_1(x)\ ,\ A_2(x) \xrightarrow{k'} A_1\big(x^{A_2@x}\big)\ ,\ A_2\big(x^{A_1@x}\big) \\ A_1(x)\ ,\ A_2(y) \xrightarrow{k_2} A_1\big(x^{A_2@y}\big)\ ,\ A_2\big(y^{A_1@x}\big) & \Longrightarrow & A_1(x)\ ,\ A_2(y) \xrightarrow{k'} A_1\big(x^{A_2@y}\big)\ ,\ A_2\big(y^{A_1@x}\big) \\ A_1(y)\ ,\ A_2(y) \xrightarrow{k_3} A_1\big(y^{A_2@y}\big)\ ,\ A_2\big(y^{A_1@y}\big) & & A_1(y)\ ,\ A_2(y) \xrightarrow{k'} A_1\big(y^{A_2@y}\big)\ ,\ A_2\big(y^{A_1@y}\big) \end{array}$$

$$\text{where } k' = \frac{k_1}{4} + \frac{k_2}{4} + \frac{k_3}{2}.$$

(b) In this second example, the signature is: $\mathcal{A} = A$, $\mathcal{S} = \{x, y\}$, $\mathbb{I} = \varnothing$, $\Sigma_\iota(A) = \varnothing$, and $\Sigma_\lambda(A) = \{x, y\}$. The sites x and y are symmetric in A whenever $k_1 = k_2 = k_3$.

Fig. 9. Examples of symmetric sites.

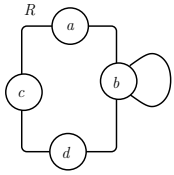
$k_1 = k_2 = k_3$. This case is more subtle, because some rules gain or loose some symmetries during the transformation. For instance, the transformation of the first rule gives four rules. One of them binds the sites x of two agents A, another binds the sites y of two agents A. The lhss of these two rules have the same number of symmetries as the one of the initial rule. Thus, their rate are divided by 4 (since the rule is rewritten into 4 rules). Another rule binds the site x of the agent A with identifier 1 and the site y of the agent A with identifier 2, and the last one binds the site y of the agent A with identifier 1 and the site y of the agent A with identifier 2. The number of symmetries in these rules is twice less as the number of symmetries in the initial rule. Thus the rate are divided by 8 (4 since the rule is rewritten into 4 rules, and 2 due to the loss of symmetries). But the two obtained rules are equivalent up to reordering and reindexing, thus, we obtain a single rule, the rate of which had been divided by 4. The transformation of the remaining rules works the same way, except that the rule which binds the sites x of two agents A and the rule which binds the sites y of two agents A both gain symmetries (the rate is divided by 4 and multiplied by 2), and the rule which binds the site x and the site y of two agents A keeps the same number of symmetries (the rate is divided by 4).

5.3 Application to the reduction of differential semantics

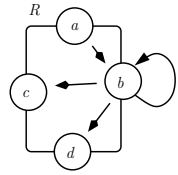
As stated by the following theorem, pair of symmetric sites induces bisimulation.

Theorem 5.3 *When two sites x and y are symmetric in A, then the relation $\sim$ which is the reflexive and transitive closure of the relation which identifies two species E and E', if and only if there exists a transposition t of states such that (i) $E' = \mathbf{subs}(\mathbf{t}, \mathbf{E})$ and (ii) t is either of the form (A, l, x, y) with $l \in \mathbb{N} \cup \overline{N}$, or of the form (B, l, z, A, x, y) with $B \in \mathcal{A}$, $l \in \mathbb{N} \cup \overline{N}$, and $z \in \mathcal{S}$, induces a bisimulation.*

Proof. This theorem is a direct consequence of Proposition 6.5 in [2]. Indeed, if the sites x and y are symmetric in the agent A, then the set of rules has the



(a) The contact map.



(b) The annotated contact map.

Fig. 10. The contact map and the annotated contact map for the Example in Section 2.3.

same differential semantics that another system satisfying the requirements of the Proposition 6.5 in [2]. We observe that the Proposition 6.5 in [2] was not dealing with binding type, but the generalization of this result is straight forward. □

Thus, we can use the framework in the Section 3.4, to define model reductions thanks to symmetric sites. We only need to pick a representative for each equivalence class.

6 Information flow-based model reduction

In this section we show how to construct an abstract/reduced semantics tracking the flow of information between different regions of chemical species. The first step is to define a family of suitable pattern components called *fragments*, that will be the basis of our abstract domain. To define our fragments, we will use an contact map (defined below) annoted with an over-approximation of the flow of information between the sites of chemical species.

6.1 Contact map and annotated contact map

The contact map associated to $\mathcal{V}$ is a summary of the bindings found in the species of $\mathcal{V}$. Specifically, the contact map (CM) is a non-oriented graph where the nodes are the pairs $(A, x) \in \mathcal{A} \times \mathcal{S}$ such that $x \in \Sigma(A)$, and the edges are the set of pairs $((A, x), (B, y))$ such that an instance of the site x in A can be bound to an instance of the site y in B in a given chemical species in $\mathcal{V}$. Therefore, any pattern projects uniquely to the contact map. The contact map for the example in the Section 2.3 is given in the Figure 10(a). As one can see, in a contact map, a site can be connected to itself (which means that an instance of the site can be bound to another instance of the same site). Moreover, some sites in the contact map may be connected to several sites, which implies a competition between two binding sites (but it does not occur in our example).

We propose to annotate the contact map with an over-approximation of the flow of information between the different regions of chemical species. The main idea is to identify the correlations between the states of the sites, which can be safely abstracted away, because they have no influence on the behavior of the states of the other sites. This way, the so-obtained annotated contact map (aCM) will be used as a symbolic description of the set of fragments of chemical species, the concentrations of which will be the variables of our reduced system.

More formally, an annotated contact map (aCM) is given by a contact map and a binary (oriented) relation over the nodes. The relation can relate two pairs (A, x) and (B, y) only if $A = B$ and $x \neq y$ or if there is an edge between (A, x) and (B, y) in the contact map. In such cases, we say that there is an arc in the aCM from the site x of A to the site y of B.

Example 6.1 An aCM for the example in the Section 2.3 is given in the Figure 10(b). This aCM should be read in the following way. In a receptor, the state of site a may influence all the state of other sites, the state of site b may influence the behavior of the state of sites c and d. Moreover, information may flow across bonds: in dimers, the state of sites a and b of a receptor can control the behavior of the state of sites b, c, and d of the other receptor.

In the Section 6.2, we define the set of fragments that is denoted by an aCM and in the Section 6.3, we give the constraints that should be satisfied by the aCM, so that it soundly summarizes the flow of information. In the Section 6.4, we define the reduced model associated to the set of fragments of a sound aCM.

6.2 Fragments

Fragments are well chosen pattern components, which can be derived from an aCM.

Let us consider an aCM. Given a pattern, we call a site instance in the pattern E a triple $(A, l, x) \in \mathcal{A} \times \mathcal{L} \times \mathcal{S}$ such that there exists an agent A with identifier l which documents the state of the site $x \in \Sigma(A)$. A *path* in a pattern E is a finite sequence of site instances $p := (A_i, l_i, x_i)_{1 \leqslant i \leqslant n}$ such that (i) for any i between 1 and $n-2$, $(A_i, l_i, x_i) \neq (A_{i+2}, l_{i+2}, x_{i+2})$; and (ii) for any i between 1 and $n-1$, either $(A_i, l_i) = (A_{i+1}, l_{i+1})$ and $x_i \neq x_{i+1}$, or the instances of the site (A_i, l_i, x_i) and $(A_{i+1}, l_{i+1}, x_{i+1})$ are bound together in the pattern E. In such a case, n is called the length of the path p, and for any i between 1 and $n-1$, $((A_i, l_i, x_i), (A_{i+1}, l_{i+1}, s_{i+1}))$ is called an arc in p. A path p in a pattern E is *compatible* with the aCM, if and only if, for any arc $((A, l, x), (A', l', x'))$ in p, there is an arc in the aCM from the site x of the agent A, to the site x' of the agent A'.

As states by the two following Lemmas, compatible paths can be composed and the image of a compatible path by an embedding is a compatible path.

Lemma 6.2 (Path composition) *If there exist two paths p_1 and p_2 in E, both compatible with the aCM and, respectively from a site instance (A, l, s) to a site instance (A', l', s'), and from the site instance (A', l', s') and a site instance $(A", l", s")$, then there exists a path in E, compatible with the aCM, from the site instance (A, l, s) to the site instance $(A", l", s")$.*

Proof. We prove the Lemma 6.2 by induction over the length of p_1. Let us write $p_1 = (A_i, l_i, s_i)_{1 \leqslant i \leqslant n}$ and $p_2 = (A'_i, l'_i, s'_i)_{1 \leqslant i \leqslant n'}$. If $n = 0$ or $n' = 0$, then p_1 or p_2 is a path in E compatible with the aCM from the site instance (A, l, s) to the site instance $(A", l", s")$; else if $(A_{n-1}, l_{n-1}, s_{n-1}) \neq (A'_2, l'_2, s'_2)$, then $(A_1, l_1, s_1), \ldots, (A_n, l_n, s_n), (A'_2, l'_2, s'_2), \ldots, (A'_{n'}, l'_{n'}, s'_{n'})$ is a path in E, compatible with the aCM; otherwise we apply the induction hypothesis with the paths

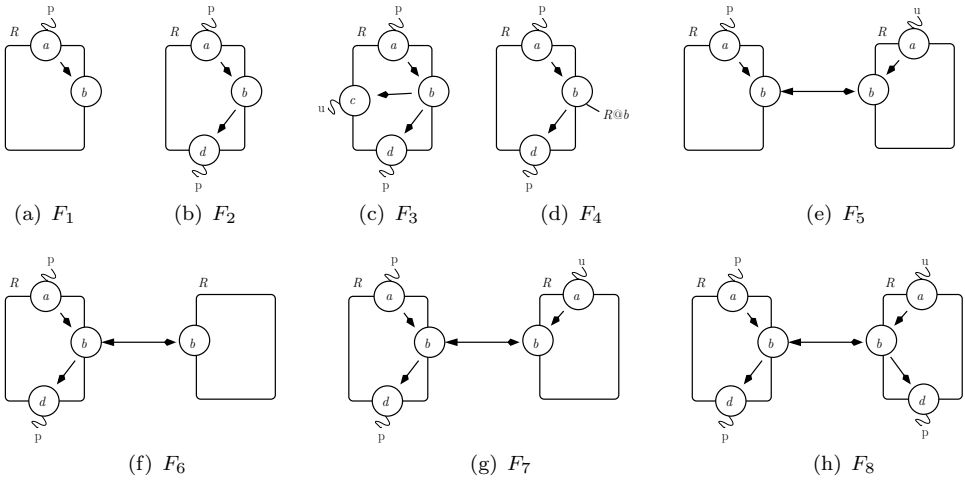


Fig. 11. Some annotated pattern components. Are they fragments ?

$(A_i, l_i, s_i)_{1 \leqslant i < n}$ and $(A'_i, l'_i, s'_i)_{1 < i \leqslant n'}$. □

Lemma 6.3 (path image) *Let ϕ be an embedding between two patterns E and E' and $(A_i, l_i, s_i)_{1 \leqslant i \leqslant n}$ be a path in E, compatible with the aCM. Then $(A_i, \phi(l_i), s_i)_{1 \leqslant i \leqslant n}$ is a path in E' which is compatible with the aCM.*

Now we define two sets of pattern components. A *prefragment* is a pattern component E such that there is a site instance (A, l, x) in E, such that, for any site instance (A', l', x') there is a path in E, compatible with the aCM, from (A', l', x') to (A, l, x). In such a case, the site instance (A, l, x) is called a *target* of the prefragment E. A *fragment* is a prefragment which is maximal for the embedding ordering: a prefragment F is a fragment whenever for any prefragment F' such that there exists an embedding between F and F', we have $F = F'$.

Example 6.4 We consider the aCM which is given in the Figure 10(b) and the pattern components which are given in the Figure 11. Among these pattern components, only F_1, F_2, F_4, F_5, F_6, and F_7 are prefragments. Yet, we can notice that F_1 can be embedded into F_2; F_4 and F_6 can be embedded into F_7. Thus neither F_1, nor F_4, nor F_6 is a fragment. On the other side, both F_2 and F_7 are fragments, since the only way to refine them is to add a site c, but in such a case, the result is not a prefragment anymore.

Given a pattern E and a state ρ over $\mathcal{V}$, we define the concentration $[E]_\rho$ of the pattern E in the state ρ, as:

$$[E]_\rho = \sum_{v \in \mathcal{V}} \frac{\mathbf{card}(\{\phi \mid \phi \text{ is an embedding between } E \text{ and } v\})}{\mathbf{sym}(E)} \rho(v).$$

We often write $[E]$ instead of $[E]_\rho$.

We define the set of abstract variables as the set of fragments of chemical species modulo isomorphisms. Formally, $\mathcal{V}^\sharp$ is a set of fragments, such that (i) for each

fragments F in $\mathcal{V}^\sharp$, there exists $v \in \mathcal{V}$ such that F embeds in v, and (ii) for any pair (F_1, F_2) of fragments in $\mathcal{V}^\sharp$, if F_1 embeds into F_2 then $F_1 = F_2$. Since $\mathcal{V}$ is a finite set, $\mathcal{V}^\sharp$ is finite as well. The set of concrete states $\mathcal{V} \to \mathbb{R}$ over $\mathcal{V}$ and the set of (abstract) states $\mathcal{V}^\sharp \to \mathbb{R}$ over $\mathcal{V}^\sharp$ are related via the abstraction function ϕ which is defined as $\phi(\rho)(v^\sharp) = [v^\sharp]_\rho$. The (pre)fragments and the abstraction function ϕ enjoy the following properties.

Proposition 6.5 (orthogonal decomposition.) *Let F be a prefragment. The concentration of a prefragment F can be expressed as a linear combination with positive coefficients of the concentration of some fragments.*

Proof. Let us define the corrected concentration $(\![E]\!)_\rho$ of a pattern E as $[E]_\rho \times \mathbf{sym}(E)$, and prove the following equivalent property: $(\![E]\!)$ can be expressed as a linear combination with positive coefficients of the corrected concentration of $(\![F_i]\!)$ of some fragments.

The proof works iteratively. At each step, a prefragment E' will be replaced with a multi-set of more refined prefragments, while preserving the overall corrected concentration, until we obtain a multi-set of fragments. If E' does not embed into a species in $\mathcal{V}$, we remove it. Otherwise E' has to be refined.

A pattern can be refined into a multi-set of patterns while preserving the overall correcting concentration, by using the following rewrite steps. We can refine the internal state of a site which misses one with any internal state in $\mathbb{I}$, or refine a binding state '?' or '$-$' with either the symbol 'ϵ' and any potential binding type (according to the CM). Moreover, a fresh site can be added in the interface of an existing agent. Lastly, if E' constains a site instance annotated with a binding type, one could replace it with a bond to an existing site (if its binding state allows it), to fresh site in existing agents, or to a site in a fresh agent.

We are left to show that, whenever a prefragment E' is not a fragment, then there always exists a rewrite step which replaces E' into a multi-set of prefragments. Only the steps which add a fresh site instance raise an issue. Thus, let us assume that no other rewrite step can apply. We consider an embedding ϕ between E' and a fragment F. We assume that there exists a target (A, l, x) in F, which has no antecedent by ϕ. Let us consider (A', l', x') a target in E'. Then we can consider a path in F compatible with the aCM from the site instance $(A', \phi(l'), x')$ to the site instance (A, l, x). The first site in this path which has no antecedent by ϕ can be aded to E' and, by construction, it is a target of the result. Otherwise, there exists a site instance (A, l, x) in E' such that (A, l, x) is a target in E' and $(A, \phi(l), x)$ a target in F and there exists a path in F compatible with the aCM from the site $(A, \phi(l), x)$ and a site having no antecedent by ϕ. The first site having no antecedent can be added to E', and (A, l, x) is still a target of the result.

Thus we have a rewriting strategies, where all intermediar steps are multi-set of prefragments. The set of prefragments which can be embedded into chemical species in $\mathcal{V}$ is finite (since $\mathcal{V}$ is finite), which ensures the termination of our iteration. □

Proposition 6.6 (divergence preservation) *We have:* $(\mathcal{V} \to \mathbb{R}) \overset{\phi}{\multimap} (\mathcal{V}^\sharp \to \mathbb{R})$.

Proof. By construction, ϕ is a linear function which maps any state over $\mathcal{V}$ to a state over $\mathcal{V}^\sharp$. Let us prove that ϕ preserves the divergence of sequences. Let us consider a sequence $(\rho_n)_{n\in\mathbb{N}}$ of states over $\mathcal{V}$ such that the sequence $(||\rho_n||)_{n\in\mathbb{N}}$ diverges. Since $\mathcal{V}$ is a finite set, there exists a variable $v \in \mathcal{V}$ such that the sequence $(\rho_n(v))_{n\in\mathbb{N}}$ diverges toward $+\infty$ (by definition, in a state ρ, $\rho(v) \geqslant 0$). Let us take an instance (A, l, x) of a site in v, we define ι as its internal state and λ as 'ϵ' if the site is free, or as its binding type $B@y$ otherwise. The pattern $E := \mathrm{A}_1(x_\iota^\lambda)$ is a prefragment. In given a mixture, the number of embeddings of E is greater than the number of embeddings of v. Moreover, by the Proposition 6.5, the number of embeddings of E in a mixture E' can be expressed as a linear combination with positive coefficients of the number of embeddings of some fragments $F_1, \ldots, F_k$ in E'. As a consequence, for at least one of the fragments, let us say F_j, the sequence $(\rho_n(F_j)_{n\in\mathbb{N}}$ diverges as well. Thus the sequence $(||\phi(\rho_n)||^\sharp)_{n\in\mathbb{N}}$ diverges as well. □

6.3 Flow analysis

In this section we define some criteria which ensure that the aCM is a sound over-approximation of the information flow. So as to make the definitions easier, we assume that the rules of our system have no side-effect, that is to say that any site which may be modified by a rule has to be documented in the lhs of the rule. More precisely, only the bonds that are written thanks to a pair of site addresses can be released, and only the agents which document the binding state of their sites with 'ϵ' or with a site address can be removed. This is not a limitation of the framework, since any rule with side-effects can be refined into a set of rules without side-effects without modifying the differential semantics of the system [13].

Some specific rules induce no flow of information. We say that a rule is *trivial*, if it is of the form $A_1\left(a^{B_2@b}\right)$, $B_2\left(b^{A_1@a}\right) \xrightarrow{k} A_1(a)$, $B_2(b)$. Thus a trivial rule release a bond without testing any other information. We could extend the class of trivial rules, but we do not do it for the sake of simplicity.

Definition 6.7 The aCM is valid with respect to a rule set $\mathcal{R}$ if it satisfies the following constraints.

(i) *direct flow:* Any path in the lhs of a (non trivial) rule r to a site instance which is modified by the rule r, is compatible with the aCM.

(ii) *indirect flow:* For any pattern component in the lhs of a non trivial rule r, there exists a site instance (A, l, x) such that any path in the lhs of the rule r to the site instance (A, l, x) is compatible with the aCM.

(iii) *hidden flow:* If a rule (trivial or not) can release a bond between two site instances (A, l, x) and (A', l', x') in a species v, and if there exist two paths $(A_i, l_i, x_i)_{1\leqslant i\leqslant n}$ and $(A'_i, l'_i, x'_i)_{1\leqslant i\leqslant n'}$, compatible with the aCM, in v respectively from (A, l, x) and (A', l', x') to a common site instance (ie such that $(A_n, l_n, x_n) = (A'_{n'}, l'_{n'}, x'_{n'})$, $(A, l, x) = (A_1, l_1, x_1)$, and $(A', l', x') = (A'_1, l'_1, x'_1)$), then there is, in the aCM, either an arc from the site x of the agent A to the site x' of the agent A', or an arc from the site x' of the agent

(a) Pullback. (b) Relative pushout.

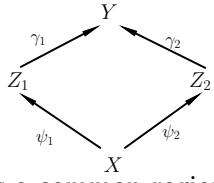
Fig. 12. Overlap.

A' to the site x of the agent A.

Intuitively, direct flows describe the flow of information between the sites that are tested (because they occur in the lhs of a rule), and the sites that are modified. Indirect flows handle with the pattern components which are not modified: the concentration of these patterns regulates the speed of rule application. Moreover, whenever a fragment contains two site instances which can potentially be bound together and there exists a rule which can release this bound, then the behavior of the fragment is not the same if the two sites are actually bound together, or not. This creates a hidden flow of information. As a consequence, we have to describe explicitly in the fragment if the two site instances are bound together, or not.

Example 6.8 The aCM in the Figure 10(b) is a valid aCM for the set of rules in the Figure 7.

So as to derive the abstract dynamic function $\mathbb{F}^\sharp$, we need to define the notion of overlap between two patterns. Our definition should be universal so that we can enumerate overlaps without over-counting them. At first glance, an overlap between two patterns Z_1 and Z_2 could be defined by two patterns X and Y and four embeddings $\psi_1, \psi_2, \gamma_1, \gamma_2$ where ψ_i is an embedding between X and Z_i, and γ_i an embedding between Z_i and Y such that the following diagram:



commutes. Intuitively X denotes a common region (which can be empty), and the existence of Y ensures that Z_1 and Z_2 are somehow compatible. Yet X and Y can be less or more refined, and thus this construction is not universal. Fortunately, whenever such a commutative square exists, it is always possible to construct an universal square where the triple (X, ψ_1, ψ_2) is a pullback and the triple (Y, γ_1, γ_2) is an idem pushout [12] (See the Figure 12). Thus, an overlap between two patterns F_1 and F_2 can be defined uniquely (up to isomorphism) by a pullback (X, ψ_1, ψ_2) and an idem pushout (Y, γ_1, γ_2), and is denoted by the tuple $(Z_1, Z_2, X, Y, \phi_1, \phi_2, \gamma_1, \gamma_2)$.We refer to [5, Section 4.B.3] for more explanations, and complete definitions and proofs.

As formalized by the following lemma, whenever two fragments overlap on a site

instance which is a target of one of the two fragments, then, the glueing of the two prefragments is also a prefragment.

Lemma 6.9 *Let $(F_1, F_2, X, Y, \phi_1, \phi_2, \gamma_1, \gamma_2)$ be an overlap between two prefragments F_1 and F_2 and such that there is a site instance (A, l, x) in X such that $(A, \psi_1(l), x)$ is a target of the prefragment F_1, then Y is a prefragment as well.*

Proof. Let us prove that, for any target (A_0, l_0, x_0) of the prefragment F_2, the site instance $(A_0, \gamma_2(l_0), x_0)$ is a target of Y. We consider a site instance (A_1, l_1, x_1) in Y. Indeed, since (Y, γ_1, γ_2) is a pushout, the site instance (A_1, l_1, x_1) in Y is either the image of a site instance in F_1 by γ_1, or the image of a site instance in F_2 by γ_2. (i) In the first case, let us introduce the agent identifier l_2 such that $\gamma_1(l_2) = l_1$ and (A_1, l_2, x_1) is a site instance in F_1. By definition of a target, there exists a path in F_1, compatible with the aCM, from the site instance (A_1, l_2, x_1) to the site instance $(A, \psi_1(l), x)$. Then by the Lemma 6.3, there exists a path in Y, compatible with the aCM, from the site instance $(A_1, \gamma_1(l_2), x_1)$ to the site instance $(A, \gamma_1(\psi_1(l)), x)$. Beside, since (A_0, l_0, x_0) is a target of F_2, then there is a path in F_2, compatible with the aCM, from $(A, \psi_2(l), x)$ and (A_0, l_0, s_0). By the Lemma 6.3, we deduce that there exists a path in Y, compatible with the aCM, from the site instance $(1, \gamma_2(\psi_2(l)), x)$ and the site instance $(A_0, \gamma_2(l_0), x_0)$. Since $\gamma_1(l_2) = l_1$ and $\gamma_1 \circ \phi_1 = \gamma_2 \circ \psi_2$, we deduce from the Lemma 6.2, that there exists a path in Y, compatible with the aCM, from the site instance (A_1, l_1, x_1) and the site instance $(A_0, \gamma_2(l_0), x_0)$. (ii) In th second case, since (A_0, l_0, x_0) is a target of F_2 then there exists a path in F_2, compatible with the aCM, from the site instance (A_1, l_2, x_1) to the site instance (A_0, l_0, x_0). Thus, since $\gamma_2(l_2) = l_1$ and by the Lemma 6.3, there exists a path in X, compatible with the aCM, from the site instance (A_1, l_1, x_1) to the site instance $(A_0, \gamma_2(l_0), x_0)$. □

The following proposition enables the computation of the activity of rules as an expression of fragments.

Proposition 6.10 (prefragment.) *Any pattern component which occurs in the lhs of a non trivial rule is a prefragment.*

Proof. The proposition 6.10 is a direct consequence of the definition of indirect flow. Let us consider a pattern component which occurs in a lhs of a non trivial rule. By definition of the indirect flow, there exists a site instance (A, l, x) such that for any path in the lhs of the rule r from a site instance to (A, l, x) is compatible with the aCM. Thus, by definition, C is a prefragment (since (A, l, x) is a target).□

A fragment cannot properly intersect a pattern component in the lhs of a non trivial rule r on a site that is modified by r, as formalized as follows.

Proposition 6.11 (left overlap.) *Let $(F, C, X, Y, \phi_1, \phi_2, \gamma_1, \gamma_2)$ be an overlap between a fragment F and the pattern component C of the lhs of a non trivial rule r. If there exists an site instance (A, l, x) in X such that the site instance $(A, \psi_2(l), x)$ is modified by the rule r, then γ_1 is an isomorphic embedding (and thus Y is a fragment).*

Proof. By the Proposition 6.10 C is a prefragment. Moreover any site instance which is modified in C by the rule r is a target. Thus, if X contains a site modified by the rule, then it contains a target of C. Thus, by Lemma 6.9, Y is a prefragment. Thus γ_1 is an embedding between the fragment F and the prefragment Y. So by definition of fragments, $F = Y$. □

Thus we can express the consumption of fragments. Conversely, the following proposition enables the computation of the production of fragments.

Proposition 6.12 (right overlap.) *Let $r := E_\ell \xrightarrow{k} E_r$ be a rule, and F be a fragment. Let $(F, E_r, X, Y, \phi_1, \phi_2, \gamma_1, \gamma_2)$ be an overlap between F and E_r. We assume that there is a site instance (A, l, x) in X such that the site instance $(A, \psi_2(l), x)$ has been modified by the rule r. We can always assume that γ_2 is a clean embedding. Thus, we can consider $r' := E'_\ell \xrightarrow{k'} E_r$, the left refinement of r by the embedding γ_2. Then if E_ℓ and E'_ℓ have the same number of pattern components, then any pattern component in E'_ℓ is a prefragment.*

Proof. [sketch] We take the notations of the theorem and assume that E_ℓ and E'_ℓ have the same number of pattern components. We consider the antecedent F' of F before the application of the rule. F' may be not connected, but any pattern component in F' intersects a pattern component in the E_ℓ on a site that is modified by the rule. Moreover, a pattern component in F' intersects at most one pattern component in E_ℓ, otherwise E_ℓ and E'_ℓ would not have the same number of pattern components. Moreover, for any bond occuring in a cycle in F without occuring in F', the two sites of the bonds occurs in the same pattern component of E_ℓ, and thus, there is a path, compatible with the aCM, from one site of the bond to the other (and conversely) in F'. Lastly, if when removing a bond in a prefragment, we obtain two pattern components, then both are fragments (at least one of the two pattern components contains the target of the initial prefragment, while one of the freed site is a target of the other pattern component). Thus, each pattern component in E'_ℓ is obtained by gluing a prefragment on a pattern component in E_ℓ on a site that is modified, thus, by Proposition 6.3 and 6.2, each pattern component in E'_ℓ is a prefragment. □

Proposition 6.13 *If a fragment F contains two site instances (A, l, x) and (A', l', x') with the respective binding states A'@x' and A@x, then no rule can release a bond between the site x of an agent A and the site x' of an agent A'.*

Proof. Let is consider a fragment F which contains two site instances (A, l, x) and (A', l', x') with the respective binding states A'@x' and A@x, we consider a target (A'', l'', x'') of the fragment F. By definition of a target, there is a path from the site instance (A, l, x) (resp. (A', l', x')) to the site instance (A'', l'', x''), thus there is a hidden flow of information, and there is an arc in the aCM either from the site x of A to the site x' of A', or an arc in the aCM either from the site x' of A' to the site x of A. In both case, one binding type can be refined into the address of a fresh or existing site. Thus we can construct a prefragment F' such that F strictly embeds into F', which contradicts the fact that F is a fragment. □

6.4 Reduced system

Following the Theorem 6.1 in [5], since the set of prefragments and the set of fragments satisfies the Propositions 6.5, 6.6, 6.10, 6.11, 6.12, and 6.13, we can derive the abstract dynamic function $\mathbb{F}^\sharp$ incrementally, by scanning the set of embeddings between the pattern components in the lhs of rules and fragments (for the consumption of fragments) and the set of overlaps between the rhs of rules and fragments (for the production of fragments).

Let r be a rule. We distinguish between two cases, when expressing the contribution of r in $\mathbb{F}^\sharp$.

- *Trivial rules.* If r is a trivial rule $A_1\left(a^{B_1@b}\right)$, $B_1\left(b^{A_1@a}\right) \xrightarrow{k} A_1(a)$, $B_1(b)$, such that there is no arc in the aCM either from the site a of the agent A to site site b of the agent B, or from the site b of the agent B to the agent a of the agent A. Then, for each embedding between either $A_1\left(a^{B@b}\right)$ and F, or $B_1\left(b^{A@a}\right)$ and F, the contribution of the consumption of F, in the rule r, via the embedding ϕ is defined as:

$$\mathbb{F}^\sharp(\rho^\sharp)(F) \stackrel{-}{=} \frac{k{\cdot}\rho^\sharp(F)}{\mathbf{sym}(E_\ell)\mathbf{sym}(F)}.$$

  Whenever $A = B$ and $a = b$, the contribution should be counted twice.

  For each embedding ϕ between either $A_1(a)$ and F, or $B_1(b)$ and F, the contribution of the production of F, in the rule r, via the embedding ϕ is defined as:

$$\mathbb{F}^\sharp(\rho^\sharp)(F) \stackrel{+}{=} \frac{k{\cdot}\rho^\sharp(F_\psi)}{\mathbf{sym}(E_\ell)\mathbf{sym}(F_\psi)}.$$

  where F_ψ is obtained by replacing with the symbol 'ϵ', the binding state of the site instance $(A, \phi(1), a)$ whenever ϕ is an embedding between $A_1(a)$ and F, or the binding state of the site instance $(B, \phi(1), b)$ otherwise.

- *Non trivial rules.* We decompose the lhs of r as a tuple a non-empty pattern components $(C_i)_{1\leqslant i\leqslant n}$. Then for any p between 1 and n, and any embedding ϕ between C_p and a fragment F, the consumption of F, in the rule r, via the embedding ϕ is defined as:

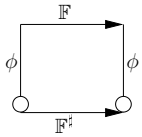
$$\mathbb{F}^\sharp(\rho^\sharp)(F)' \stackrel{-}{=} \frac{k{\cdot}\rho^\sharp(F)}{\mathbf{sym}(E_\ell)\mathbf{sym}(F)} \prod_{i\neq p}[C_i].$$

  Moreover, for any overlap $(F, E_r, X, Y, \phi_1, \phi_2, \gamma_1, \gamma_2)$ between a fragment F and the rhs of the rule r, if the number of pattern components in the lhs of the left refinement of r by the embedding γ_2 is the same as the number of pattern components in the lhs of r, then the production of F, in the rule r, via the overlap $(F, E_r, X, Y, \phi_1, \phi_2, \gamma_1, \gamma_2)$ is defined as:

$$\mathbb{F}^\sharp(\rho^\sharp)(F) \stackrel{+}{=} \frac{k}{\mathbf{sym}(E_\ell)\mathbf{sym}(F)} \prod_i [C'_i].$$

  Otherwise the overlap has no contribution.

Theorem 6.14 *If $\mathcal{V}^\sharp$ is the set of fragments which is denoted by a valid aCM, then, the following diagram:*

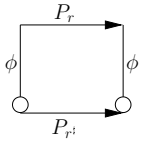


is a model reduction.

Proof. The proof that is given in the Section 6 of [5] applies, providing the fact that we replace the word 'subfragment' with the word prefragment. □

6.5 Combining fragments and symmetries

Now we wonder when we can use the potential symmetries among sites, so as to reduce further the number of fragments. Thanks to the framework that we have proposed in the Section 3.5, we propose to quotient the set of chemical species in $\mathcal{V}$ and the set of fragments $\mathcal{V}^\sharp$ by some equivalence relations which identify chemical species and fragments upto permutation of symmetric sites. Yet, this can only be done if we can provide two idempotent functions $r : \mathcal{V} \to \mathcal{V}$ and $r^\sharp : \mathcal{V}^\sharp \to \mathcal{V}^\sharp$, mapping respectively each species and each fragment to a representative, and such that the following diagram:



commutes.

Given two symmetric sites x and y in an agent A, we consider two cases. If there is an arc in the aCM from the site x in A, to the site y in A, then there is also an arc from the site y in A to the site x in A in the aCM. Thus, in a fragment, whenever an agent A documents the site x, the site y is documented as well (and conversely). In such a case, it is always possible [2] to choose the functions r and $r^\sharp$ which abstract away the difference between the site x and the site y in agents A, so that the diagram commutes. Otherwise, there is no way to define the mappings r and $r^\sharp$ such that the diagram commutes, but, since no agents A in a fragment documents both x and y, there is no need to abstract the difference between these two sites.

7 Conclusion

We have proposed a formal framework for reducing the differential semantics of rule-based models. This framework combines two abstractions: we use the flow of information to detect useless correlations and the pairs of site having the same capabilities of interaction to abstract away any distinction between these sites. The initial semantics and the reduced one are formally related by Abstract Interpretation.

In future works, we will implement this framework within the OpenKappa platform (downloadable at kappalanguage.org). Then, we will address the combinaison of the reductions based on the detection of useless correlations (as in this framework), and the ones based on the detection of invariants (as in [9,8]). On the theorical side, we are looking for a semantics definition of the flow of information (based on the set of ground reactions induced by a rule-based model), both for the stochastic semantics and for the differential one. Then we will describe the abstractions of the flow of information which is used in this paper and in [7,5,10,9,8] as a hierarchy of abstractions of this semantics definition of the flow of information.

References

[1] Blinov, M. L., J. R. Faeder and W. S. Hlavacek, *BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains*, Bioinformatics **20** (2004).

[2] Camporesi, F., J. Feret, H. Koeppl and T. Petrov, *Combining model reductions*, in: *MFPS XXVI*, ENTCS **265** (2010).

[3] Conzelmann, H., D. Fey and E. D. Gilles, *Exact model reduction of combinatorial reaction networks*, BMC Systems Biology **2** (2008).

[4] Cousot, P. and R. Cousot, *Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints*, in: *POPL 77*.

[5] Danos, V., J. Feret, W. Fontana, R. Harmer and J. Krivine, *Abstracting the differential semantics of rule-based models: exact and automated model reduction*, in: *LICS '2010*.

[6] Danos, V. and C. Laneve, *Core formal molecular biology*, TCS **325** (2003).

[7] Feret, J., V. Danos, J. Krivine, R. Harmer and W. Fontana, *Internal coarse-graining of molecular systems*, PNAS **106** (2009).

[8] Feret, J., T. Henzinger, H. Koeppl and T. Petrov, *Lumpability abstractions of rule-based systems*, in: *MeCBIC 10*, EPTCS.

[9] Feret, J., H. Koeppl and T. Petrov, *Stochastic fragments: A framework for the exact reduction of the stochastic semantics of rule-based models*, IJSI To appear.

[10] Harmer, R., V. Danos, J. Feret, J. Krivine and W. Fontana, *Intrinsic information carriers in combinatorial dynamical systems*, Chaos **20** (2010).

[11] Ince, E. L., "Ordinary Differential Equations," Dover Publications, Inc., New York, NY, 1956.

[12] Leifer, J. J. and R. Milner, *Deriving bisimulation congruences for reactive systems*, in: *CONCUR 00*, LNCS **1877**.

[13] Murphy, E., V. Danos, J. Feret, R. Harmer and J. Krivine, *Rule based modelling and model refinement*, in: *Elements of Computational Systems Biology*, Wiley Book Series on Bioinformatics, 2009 .

[14] Petrov, T. and H. Koeppl, *Maximal Reduction of ODE Semantics of Rule-based Models: Syntax-Independent Setup*, in: *International Workshop on Computational Systems Biology*, 2010.