Daniela Lepri

A Formal Semantics for Molecular Interaction Maps

Master Thesis

University of Pisa
February 2008

Università di Pisa
Facoltà di Scienze Matematiche, Fisiche e Naturali

Corso di Laurea Specialistica in Informatica

A Formal Semantics for Molecular Interaction Maps

Relatori

Prof. Roberto Barbuti	....
Dott. Paolo Milazzo	....

Candidato

Daniela Lepri	....

Sessione di Laurea Febbraio 2008
Anno Accademico 2006/2007

Abstract

In the present work, we describe a possible formal semantics for Molecular Interaction Maps (MIMs), which are standard diagrams, used by biologists to depict interactions at molecular level within a cell environment. First, we describe MIM notation in details, stressing two of the possible interpretations for a map: explicit and combinatorial. Then, we describe the Calculi of Looping Sequences (CLS), a family of formal languages, which models biological systems, whose semantics is a transition systems. Finally, we give a possible formal semantics for MIMs, translating them into some CLS. We give a detailed formal semantics for the explicit interpretation of a MIM, through translation into Stochastic CLS+ (Stochastic CLS *Plus*): we formally define a possible intermediate encoding of MIMs and we provide a translation of this encoding into a stochastic CLS+ model; finally, we show how we can apply our formal semantics to the EGFR Signalling Pathway explicit map, encoding part of it into a CLS+ model. We give an informal semantics for the combinatorial interpretation of a MIM, through a possible translation into Stochastic LCLS (*Plus* extended): we study how complexes can be represented into LCLS and the challenge of managing cell membranes.

Acknowledgements

I would like to thank the people who made this thesis possible.

I thank my supervisor Roberto Barbuti for his assistance and patience with this thesis work and for introducing me to the world of computer science some years ago. He helped to make university fun for me, making me enthusiast about the degree course I chose.

I thank my supervisor Paolo Milazzo, for many insightful conversations during the development of the ideas in this thesis, and for helpful comments on the text.

I am grateful to my family, who patiently waited for this degree, supporting me in both nice and difficult moments. I warmly thanks all of my relatives, who proudly encouraged my studies.

A sincere thanks goes to my cousin Angela, who helped me finding my way during my first years in Pisa: I enjoyed very much the time we spent together in Pisa

I thank all of my friends, which tolerated my periods of emotional tension, especially during these last months.

In particular I am deeply grateful to Nertila, who has always been close to me and supportive, sharing my joy and sadness and helping me to find motivations for working harder: I owe my loving thanks to you.

I warmly thank Sabina for keeping my spirits up with her bright vitality. I wish to thank Stine, my wonderful friend, for all the nice travels we have done together: I am very happy and lucky that I met you. I also wish to thank Gio for his wise advices. I also thank Barbara, Jenni, Giovanni, and all the nice persons I met during these years. Finally, a special thanks goes to Peter: I hope I can soon challenge you at badminton and start a nice cooperation with you.

Contents

List of Figures

List of Tables

Introduction

"The greatest challenge today in cell biology is the accurate and complete description of complex systems. The next task is to assemble mathematical models that capture the key system properties".

E. O. Wilson

Understanding the mechanism underlying the biological behavior of cells is a major challenge. The function of a cell is essentially governed by interactions between molecules of a certain species. The underlying order of such interactions is very hard to capture, due to their multitude and the fact that molecules can continuously move around a cell, cross membranes, and interact with other molecules in several ways, depending on which other components are present in their "neighbourhood". This makes tracing of metabolic pathways (that is, sequence of interactions) extremely difficult.

These molecular interactions can be organized in a kind of network diagram much more complicated than a usual electronic circuit one, namely a *Bioregulatory Network*. Moreover, the information about possible interactions and components is becoming more detailed within these years, increasing the need for a complete and unambiguous standard graphical notation for building diagrams of Bioregulatory Networks.

Kurt Kohn was the first trying to give rigorously a graphical notation for bioregulatory networks, namely the Molecular Interactions Map (MIM) notation, which he presents in [1]. Other researchers tried to improve Kohn's notation adopting different approaches, some of which are modifications of MIM notation: [2, 3, 4, 5, 6, 7, 8, 9]. K. W. Kohn *et al.* describe in [10] the MIM notation formally and strength and weakness of some of the alternative notation that have been proposed: among these, Kitano's Process Diagrams [4, 5, 6]), CADLIVE software suite [2], automated diagrams of Cook *et al.* [3], and BIOCARTA's connection diagrams (http://www.biocarta.com).

MIM diagrams depict molecular species (Proteins, DNA, RNA, other molecules, multimolecular complexes and multi–domain molecular species as well) as nodes and interactions among them as connection lines, whose meaning depends on the line type and the kind of its end symbols, usually arrowheads. Interactions include reactions (binding, modifications, stoichiometric conversion, DNA transcription, etc.), which yield the modification or creation of some molecular complex, and contingencies (enzyme catalysis, stimulation and inhibition of biochemical reactions, etc.), which affect the behavior of reactions. Moreover, it is possible to define a spatial positioning of the various components, through sketching of membranes.

Process diagrams proposed by Kitano in [4] are quite similar to MIM diagrams. Kitano tried to maintain the same symbology used by Kohn in representing interactions. A main difference is in the layout of information about a certain species, like binding sites, domains and others: in a MIM this kind of information stays outside the species node or, when the map could be too over crowded, this information is included in apposite annotation tables; while, in process diagrams, this information is usually inside the node. Kitano *et al.* also developed CellDesigner [7], a computer-aided design (CAD)-like software to depict process diagrams. In [11, 12] Kohn *et al.* compare the MIM and process diagram notations in detail and consider where each may be advantageous.

At present days, it appears that MIM notation could be suitable as a standard, since it has been used successfully to create maps of several bioregulatory netwoks ([13, 14, 15, 16, 17, 18, 19, 12], http://discover.nci.nih.gov/mim/).

Describing all the various interactions in a biochemical network in a single diagram, allows tracing of pathways, that is sequences of interactions, possibly with the aid of computer simulation. In [11], Kohn *et al.* remark one of the main peculiarity of MIMs: the possibility of interpreting interaction symbols appearing in a MIM in different ways, depending on the map application. The chosen interpretation changes radically the set of pathways depicted in the map. The three different possible interpretations for a MIM are:

- *explicit*, aimed at defining specific models for computer simulation;
- *combinatorial*, aimed at diagramming combinatorially complex models;
- *heuristic*, aimed at organizing available information about a network's molecular interactions.

Depending on the intended application, one must specify the way in which a MIM is to be interpreted.

The meaning of MIM symbols is usually clear and simple to understand, however, some ambiguities can arise in the use of a particular class of interaction symbols, namely contingency symbols. The lack of mathematical interpretation for these symbols, led Kohn *et al.* to the conclusion that they should be avoided in maps defined for computer simulation. Hence, explicit maps should only use a subset of the MIM symbols.

In this work, we show how, giving a formal semantics to contingency symbols, we can use them in diagrams aimed at computer simulation.

Explicit maps describe a determined set of reactions and hence, can be easily simulated through ordinary differential equations (ODE), while, combinatorial maps implicitly depict a possibly infinite variety of reactions, thus, they are not easy to simulate through standard mathematical ways.

In this work, we propose few guidance on how combinatorially complex models could be formalized, keeping the encoding dimension proportional to the map dimension, thus, the formalization shall model implicitly all possible reactions, included those which are not explicitly shown in the map.

In the present work we will provide a comprehensive formal semantics for the explicit interpretation and we will cover in a less formal way the main aspects of giving a formal semantics for the combinatorial interpretation, taking into account both qualitative and quantitative aspects of a MIM model. The formal semantics is given by translation into an adequate *formalism*, which is able to descrive the biological components and interactions in a molecular interaction map.

Many formalisms, originally developed to model systems of interacting components, have been applied to Biology. For instance Petri Nets [20] and π-calculus [21, 22]; some new formalisms have been proposed to describe biomolecular and membrane interactions, like Brane Calculi [23], Beta Binders (an extension of π–calculus, [24]) and a probabilistic model for molecular systems developed by Barbuti *et al.* [25]; finally, P Systems ([26] they are termed as the last name of their creator Gheorghe Păun), are computational models, originally based upon the membrane architecture of a biological cell, which have been only eventually applied to biological system, and whose variations led to the formation of a branch of research known as “membrane computing”.

We use formalisms in that family of calculi termed *Calculi of Looping Sequences*, developed by Barbuti R., Maggiolo–Schettini A., Milazzo P. and Troina A. at the University of Pisa. This family of calculi includes several formalisms: Full–CLS, CLS, CLS+, LCLS, Stochastic CLS [27, 28, 27, 29, 30].

The *Calculus of Looping Sequences* (CLS, [27]) is a calculus based on term rewriting, suitable to the description of biological systems. It provides means for describing molecules into a membrane structure and the possible evolution of such a system, moreover it has fairly simple syntax and semantics. The semantics of CLS is a transition system, where states are given by terms and transitions from a state to another are given by rewrite rules. Describing biological interactions in terms of rules, avoids the combinatorial explosion encompassing differential equations. Moreover, in contrast to the synchronicity of differential equations, rules operate in concurrency, thus providing a more appropriate representation of cell components behavior.

For each interpretation of a MIM, explicit or combinatorial, we chose a calculus of looping sequences that could suit best the its peculiarities: concerning explicit maps, we chose a Stochastic extension of CLS+, while, for combinatorial maps, we choose a Stochastic extension of LCLS (*Plus* extended). A Stochastic extension adds the ability of modeling quantitative aspects of biological systems, such as time and probability of having a certain evolution of the system.

In giving a formal semantics form MIMs, we first encode them into a formal intermediate representation, through the definition of some structures coding each component in a MIM (species, membranes, reactions, contingencies, etc.). Then we define translation functions that, given a MIM in its intermediate encoding, produces the appropriate CLS model.

Thesis structure

The thesis is structured as follows.

- In Chapter 1, after some background notions about Biology and Bioregulatory Networks, we describe the Molecular Interaction Map notation in details. We show the basic principles on which MIMs rely and their syntax for depicting elementary and complex species, and interactions between them, both reaction and contingencies. Then, we focus on their meaning, stressing the differences of the three possible interpretations: explicit, heuristic and combinatorial. We concentrate on explicit and combinatorial maps, since they will both be object of our formalization: we make a short comparison, evidencing their peculiarities and the aspects we should take into account in our formalization. We also recall quantitative aspects of the system represented by a map: in particular, we show how speed and time of each reaction can be simulated by a possible stochastic formalism, using the classical Gillespie's Algorithm [31] for stochastic simulation of chemical reactions. We make some considerations on how membranes can be depicted in a MIM: we discuss how interactions can be interpreted in different ways (locally and globally), in presence of a membrane structure, and we motivate our choice of making interactions local to a certain spatial position.

Finally, we show an example of MIM diagram, so that the reader can take a little familiarity with the notation.

- In Chapter 2, we describe in details the Calculi of Looping Sequences. First, we motivate our choice for this family of calculi as means to give a formal semantics to MIMs. We make a brief panoramic over the various calculi included in this family and we proceed recalling them in more detail. We start describing CLS [32, 27], since other calculi can be considered an extension of this one. We present its fairly simple syntax and semantics formally: in this calculus, terms are constructed by using operators of sequencing, parallel composition and looping–and–containment. CLS poses a syntactical constraint on possible commutativity on looping sequences, by which only a sequence can be enclosed in a loop (and hence be on a membrane surface). Before explaining the details of its extensions, we make the reader familiar with the usage of CLS as an abstraction of biomolecular systems, showing general guidelines to model biomolecular entities and events, as given in [27]. We describe CLS extensions, starting from CLS+ [27], which introduces a form of commutativity on looping sequences, by which a parallel composition of sequences can be enclosed in a loop, and thus allows modeling membranes in a more natural way. Then, we describe in a less formal way the the "linked" extension of CLS, namely the Calculus of Linked Looping Sequences (LCLS, [33, 30]), which allows creating links (bindings) between individual elements of different sequences and hence, can be used to model interactions at the domain level (a complete formal semantics for LCLS can be found in appendix A). Finally, we recall the Stochastic extension of CLS [28], which is suitable to describe quantitative aspects of biological systems such as the frequencies and the probabilities of events. The extension is obtained by combining rate constants to rewrite rules of CLS, and by incorporating the stochastic framework of Gillespie's algorithm in the semantics of the formalism.

- In Chapter 3, we introduce our formal semantics for Molecular Interaction Maps. First, we give the reader a panoramic view on how we proceed in our formalization. We stress the differences and common points of the two approaches we follow, for the explicit and combinatorial interpretation of a map, respectively. Before starting with the explicit and combinatorial formalizations, we give some guidance on how we interpret interactions symbols, focusing on contingency symbols. Contingencies have been a notorious question in standard simulation approaches of MIMs, and usually excluded from maps aimed at computer simulation, since they are considered source of ambiguities in the meaning of a map. We show how, giving a clear semantics to contingencies, allows them to be used in depicting maps aimed at simulation. We proceed introducing a comprehensive formal semantics for explicit maps, by translating them into Stochastic CLS+ (an hypothetical stochastic extension of CLS+). After having encoded all the components of a map (species, membrane structure, reactions and contingencies), we proceed by defining a formal translation of this intermediate representation into a Stochastic CLS+ model: an initial term will represent the initial configuration and topology of the map, while a set of rewrite rules will model all the stochastic events which can affect the evolution of this initial configuration. After having defined a formal semantics for explicit maps, we present an example of its application to part of the well–note Epidermal Growth Factor (EGF) signalling pathway: following our semantics, we translate part of an official explicit map for this pathway into a stochastic CLS+ model and we show a possible execution of the resulting model. We point out how a catalysis contingency can be included in the model, in accordance to the meaning of the explicit map. Finally, we provide a less formal semantics for combinatorial maps, by translation into Stochastic LCLS+ (an hypothetical LCLS *Plus* and stochastically extended). We define formally how an initial term can be extrapolated from a given map, then we give less formal guidelines on how defining a proper set of rewrite rules for a given map. We give the basic idea behind the combinatorial representation of complexes, keeping the encoding proportional to the map dimension. We point out the delicacy in managing membranes, due to the possibility of translocate a complex from a place to another, and we give a possible solution to address this question. We remark how combinatorial maps can be formalized in a fairly clear manner in the absence of a membrane structure.

- In Chapter 4, we give some conclusions, discussing related works and eventual future work. Concerning related works, we make a brief comparison of our work with a similar work made with β–binders [34] and a recent work presented by Fonda S. in his Master Thesis [35].

Chapter 1

Molecular Interaction Maps - MIMs

1.1 Brief Biological Background

In a *chemical reaction*, substances, known as *reactants*, interact with one another to create new substances, called *products*. A chemical reaction can be represented by a chemical equation, in which the chemical symbols on the left stand for the reactants, and those on the right are the products. An arrow points from reactants to products and optionally, a *kinetic constant* (sometimes named *reaction constant* or *rate constant*) can be scripted on the arrow, quantifying the "speed" of the reaction. For instance:

$$nA + mB \underset{k_2}{\overset{k_1}{\rightleftharpoons}} C$$

denotes a *reversible* reaction where n molecules of substance A and m molecules of substance B can react together to create one molecule of substance C as product, at a speed rate depending on the kinetic constant k_1 and the concentrations of the reactants, and conversely C can be converted into the respective amount of A and B with k_2 as kinetic constant; n and m are named *stoichiometric coefficients*. Irreversible reactions are denoted by a single arrow $\rightarrow$.

A certain energy threshold (*activation energy*), must be crossed before a reaction can occur. A reaction can be speeded up in various way, for instance, increasing the concentration of reactants or increasing the temperature at which the reaction takes place or introducing a *catalyst*, a substance that speeds up a reaction without being consumed in the reaction. The rate of a reaction follows the *mass action law*, by which "the rate of a chemical reaction for a uniform system at constant temperature is proportional to the concentrations of the substances reacting"[1].

[1] "mass action law". McGraw-Hill Dictionary of Scientific and Technical

Biochemistry is the study of chemistry of living organisms and life processes. It involves the quantitative determination and structural analysis of the organic compounds that make up cells, focusing on the big amount of interrelated chemical reactions happening within a cell. A series of chemical reactions occurring within a cell is termed *metabolic pathway.*

Molecules within a cell can interact with each other in different ways: let‘s see a couple of examples of possible biochemical reactions using the equation notation:

$A + B \to A{:}B$ molecules A and B bind together, forming a new complex molecule, called a *dimer*, named $A{:}B$; this reaction is termed *complexation*, the reverse reaction is termed *decomplexation*; if a molecule of a certain species binds to a molecule of the same species, the complex is named *homodimer*;

$A \to pA$ a phosphate p is added to a molecule A, this *modification* is termed *phosphorylation.* In an equivalent manner, other modifications can apply to a molecule, in particular to proteins, such as acetylation, ubiquitination, etc. A modification usually *activates* an otherwise metabolically inert molecule, that, once modified, becomes ready to interact with other molecules.

Cells are the basic functional unit of which all living things are composed. A cell is bounded by a *membrane*, that enables it to exchange certain materials with its surroundings. Membranes, such as the plasma membrane and nuclear membrane, appear also inside a cell, dividing the cell into compartments. A cell contains various molecules which can be positioned also onto membranes, thus acting as “interface” between different compartments.

Among the molecules appearing in a cell, there are nucleic acids, which in the form of DNA (deoxyribonucleic acid) and RNA (ribonucleic acid) control cellular function and heredity, and proteins, which are fundamental components of a living cell. *Proteins*, which are much of a living organism’s dry weight, are complex molecules built from chains of amino acids, whose order and structure determines the protein function: in facts, a protein carries out its function by binding to a specific molecule and the particular structure of a protein determines which kind of molecule can fit together with the protein. The region of the protein responsible for binding another molecule is termed *binding site* or *domain.* A protein can also bind to a membrane or even being encapsulated in a membrane. Different proteins are synthesized in a cell, according to instructions given by DNA and carried out by RNA and other proteins. Proteins have a wide variety of structural and functional roles, among them: acting as primary

Terms. McGraw-Hill Companies, Inc., 2003. Answers.com 22 Dec. 2007. http://www.answers.com/topic/mass-action-law building blocks of tissues, immunizing the system (antibodies), regulating metabolism (hormones), catalyzing chemical reactions (enzymes), transporting gas (hemoglobin) and other important functions.

A particularly important type of protein are *enzymes*, which serves as catalysts in a chemical reaction: they speed up a chemical reaction without being consumed in the reaction. Thus, enzymes facilitate chemical reactions without raising temperatures or increasing the concentrations of substances: it is as if the activation energy barrier of a reaction would be reduced in presence of enzymes. Being a protein, each type of enzyme can interact with only one particular type of substance, termed a *substrate*. The binding site of the enzyme (termed *active site*) fits together with the substrate, forming an enzyme–substrate complex.

Let‘s have a closer look to how an enzyme works: suppose a substrate molecule needs to be broken apart (think about digestion of carbohydrates), an enzyme with the correct shape arrives and attaches itself to the substrate molecule; the formation of this bond causes the breaking apart of other bonds within the substrate molecule; after finishing its job, the enzyme looks for another molecule to be catalyzed. An example of class of enzymes are the kinases, which catalyze the phosphorylation of their substrates.

Bioregulatory Networks: major features

Bioregulatory Networks look at interactions within a cell at molecular level. Emergent behavior of the cell and its major components can be observed in consequence of these molecular interactions. Bioregulatory Networks are characterized by unique features. The followings are only some of them, that would be likely taken into account by a representation diagram:

- Molecules can be of different species, thus we have different protein species, DNA fragments, and other kind of species.
- Molecules of a certain species can bind to another one of the same or different species, forming a multimolecular complex of a new species; the resulting complex has new interaction capabilities.
- Molecules are subject to modifications (phosphorylation, acetylation, etc.); a modified molecule has new (interaction) binding or enzymatic capabilities.
- enzymes can possibly affect the interaction between two molecules or complexes.
- A molecule can have multiple modification sites and hence be subject to different modifications.
- Protein molecules can have several domains (intra–molecular domains) and each domain has its own interaction capability.

- different intra–molecular domains can interact with each other.

Indeed, the interaction (**binding, modification, enzymatic**) capabilities of a molecule depend on the functional state of the molecule at a given moment. This functional state is determined by several different events, like: possible bindings to other molecules, the functional state of these molecules to which it is bound, the functional state of all the other molecules surrounding the given molecule. All these dependencies and complex patterns of modifications and creations of complexes are some of the reasons for the high complexity of the system represented by a bioregulatory network.

1.2 Molecular Interaction Maps

In this section we step through the Kohn's notation for Molecular Interaction Maps (MIMs), following the directives given by K. W. Kohn *et al.* in [10], which is at present days the most recent and exhaustive work available in literature. We will avoid the use of abbreviations in MIM notation, since indeed, in order to be formalized, they would be made explicit anyway.

1.2.1 General principles

A MIM essentially depicts some molecular species (multimolecular and multi–domain molecular species as well) and the interactions among them: these interactions are represented by connection lines, the meaning of which depends on the line type and the kind of its end symbols, usually arrowheads.

The basic principle on which MIMs rely is that a named molecular species can appear only once in a diagram, except for small or ubiquitous molecules. To this end, note that a MIM does not tell anything about the number of molecules of a certain species present at a given moment (the concentration of that species): a molecular species depicted in the map, represents the set of all the molecules of that particular species, which can be empty. Indeed the map is “state–less”. In this sense, MIM diagrams are *canonical*: they are general, not restricted to a particular state or particular cell, they show interactions that *can* occur, if certain molecules are colocalized in a certain place at a certain time; moreover, a MIM does not show any order of events.

1.2.2 Molecular species symbols

Two classes of molecular species are to be represented:

Elementary species, mainly consisting of monomolecular species, which can possibly have more than one interaction domains and/or more than one modification sites, or DNA strands;

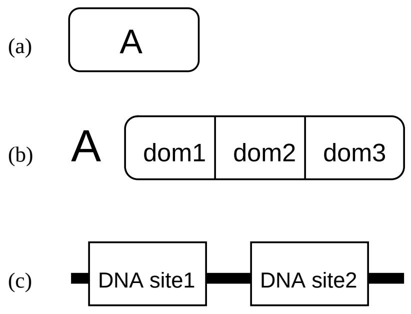

Figure 1.1 Elementary species in MIM

Complex species, consisting of combinations of elementary species (multimolecular species) and modified elementary species.

Elementary species

Figure 1.1 shows how elementary species are depicted in MIM notation. A named elementary species is represented with a rounded box, containing its name (Fig. 1.1.(a)).

An elementary molecular species can contain several interacting domains and/or several modification sites. A multi–site species can thus be modified more than once. Sometimes site names are explicitly written at the point where the interaction line meet the species (inside or outside the species box), with the convention that two modification lines pointing at two different points around the box are considered to be affecting two different sites. Thus, when two modification lines are pointing to the exactly same point, they are competing for the same site.

In case of multi–domain species, the respective rounded box will contain the domain names, separated by vertical lines, and the species name is written adjacent to the left end of the box (Fig. 1.1.(b)).

An interaction line can point to the specific domain or site, or when this information is unknown, can point to the species name written adjacent to the box. Notice that each single domain has the same interaction possibilities of a single molecular species.

In case the molecular species is a DNA strand composed by several DNA sites, this is represented by a thick line with several sharp rectangles on it, one for each DNA site, containing the name of the DNA site (Fig. 1.1.(c)).

Complex species

Figure 1.2 shows how complex species are depicted in MIM notation. A complex molecular species, resulting from an interaction, is depicted as a

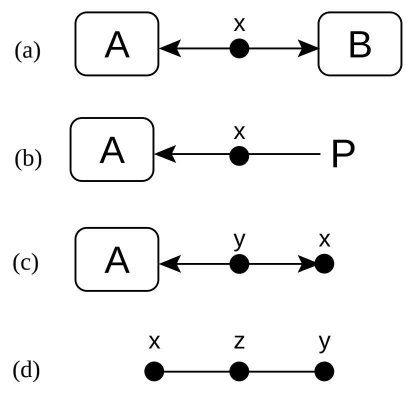

Figure 1.2 Complex species in MIM

node (bullet) on the respective interaction line. For instance, the complex species composed by the binding of two elementary species (called dimer) is represented by a node on the binding line (Fig. 1.2.(a)).

Similarly, a modified species is represented by a node on the modification line (Fig. 1.2.(b)).

Nodes are given names like $x, y, z...$ for sake of simplicity. Several nodes can be placed on the same line, representing exactly the same molecular species, but possibly different instances.

An isolated node connected to a species with an interaction line represents another instance of that species. This is the so–called “isolated node” convention. For instance, in Fig. 1.2.(c) node x is another instance of species A. In order to disambiguate some particular situations (see Fig 1.12 for an example of this usage), a short line pointing to the species represented by the isolated node can be used.

A node on a line without arrowheads which connect two species represents a state where both species at the ends of the line are present. Fig. 1.2.(d) is an example of usage of this “state–combination” symbol. Notice that nodes have the same interaction possibilities of named species.

1.2.3 Interaction symbols

Interactions can be divided into two classes:

***Reactions*,** which affect molecular species;

***Contingencies*,** which affect reactions or other contingencies.

Note again, that an interaction symbol represents a *possible* interaction, that can happen if certain state conditions hold.

A kinetic constant k can be associated to each interaction in the map, denoting the “speed” of that interaction: an interaction with an higher kinetic constant is more likely to happen.

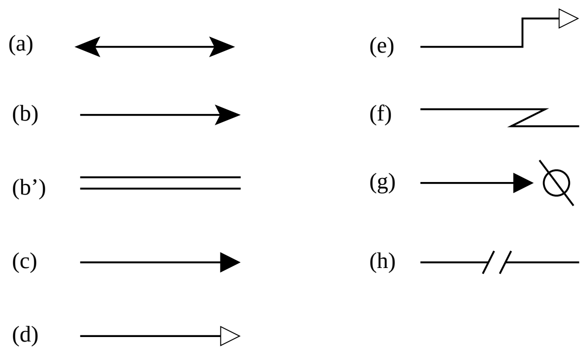

Figure 1.3 Reaction symbols in MIM

Note that a non–covalent binding, that is a reversible binding, has a pair of kinetic constants associated with it: the first one for association and the second one for dissociation.

Contingencies can then be seen as a replacement of this kinetic constant with a new one. Contingencies could yield some ambiguities in the meaning of a map, thus Kohn *et al.* suggest not to use them in maps that should be used as input for computer simulation.

Reaction symbols

The set of reaction symbols is listed in figure 1.3 and their meaning is the following:

(a) Non–covalent binding denotes the reversible binding of the two pointed species: a molecule of the first species can bind to a molecule of the second species, forming a compound. Two species joined in a non–covalent binding can eventually dissociate again.

(b) Covalent modification denotes the covalent modification of the pointed species; the modification type (phosphorylation, acetylation, ...) is written at the tail;

(b') Covalent binding denotes a covalent bond of the two connected species; (Note that this symbol will eventually be used also for Covalent modification, since the second lacks of symmetry)

(c) Stoichiometric conversion denotes the conversion of a species at the tail of the arrow, called reactant, into a corresponding number of product species; in other words, the species written at the tail of the arrow disappears, while the pointed one appears; note that this symbol, when pointing to a bullett, represents the **translocation** of the species at the tail to the pointed point.

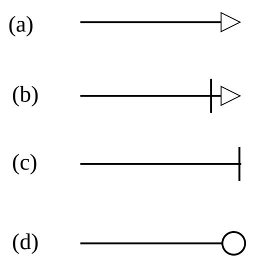

Figure 1.4 Contingency symbols in MIM

(d) Lossless production it is the same as (c), but without loss of the reacting species.

(e) Transcription denotes lossless production by transcription, like mRNA production.

(f) Cleavage of a covalent bond denotes the possibility of a covalent bond at the head to be broken by the presence of the species at the tail. (Note that this symbol points from a species to another reaction symbol).

(g) Degradation denotes a stoichiometric conversion to debris, that is the loss of the species at the tail.

(h) Reaction *in–trans* when this gap symbol is present on a reaction line connecting the same species, it denotes a reaction *in–trans*, that is a reaction between two different molecules of the same species.

Contingency symbols

the set of contingency symbols is listed in figure 1.4 and the meaning of each contingency symbols is:

(a) Stimulation the presence of the species at the tail makes the pointed interaction more likely to happen;

(b) Requirement the presence of the species at the tail is required for the pointed interaction to happen;

(c) Inhibition the presence of the species at the tail decreases the possibility for the pointed interaction to happen;

(d) Catalysis when the species at the tail is present (an enzyme), the pointed interaction can happen more easily (in a faster manner).

The difference between stimulation and catalysis is noticeable when the pointed interaction is a non–covalent binding. In this case stimulation has the double meaning of stimulating the binding, when the two species are not

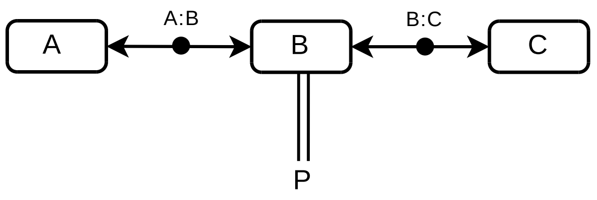

Figure 1.5 Example to illustrate the “explicit”, “combinatorial”, “heuristic” interpretations of MIMs

associated yet, and inhibiting the dissociation, when the two species are bounded, while a catalysis pointing to a non–covalent binding stimulates both association and dissociation of the species of interest.

1.2.4 Three Interpretations of MIMs

In [11], Kohn *et al.* clarify how a MIM should be interpreted. They distinguish three kinds of alternative interpretations for one MIM: *explicit*, *combinatorial* and *heuristic*. Each interpretation is suited to a different purpose, depending on the intended application. One should specify the way in which a MIM is to be interpreted. The three interpretations differ in how interactions between **indirectly** connected species should be considered.

Figure 1.5 shows a small example of MIM, which explicitly shows the bindings between **A** and **B**, yielding **A:B**, the binding between **B** and **C**, yielding **B:C** and the possible phosphorylation of **B**, yielding **pB**. Some questions spontaneously arise: should we consider as possible binding of complex **A:B** with **C**, yielding **(A:B):C**? Should we consider the binding between **pB** and **A**, yielding **A:pB**, possible or not?

Here comes the difference between the three possible interpretations. Table 1.1 shows which complex species are taken into account by each interpretation.

Explicit

A MIM, in its explicit interpretation, depicts explicitly each possible reaction: an interaction line applies only to the molecular species directly connected to it. Metabolic pathways are easy to find in this interpretation: the order of bindings is depicted explicitly in a sequential way, thus, this type of MIM defines the reaction paths for a particular model. In Figure 1.5, the explicit interpretation allows only the formation of **A:B**, **B:C** and **pB**.

Explicit maps can be built using only a subset of MIM symbols: all contingencies symbols may, on the whole, be represented by a set of reaction symbols (see [10] for details). For instance, **enzymatic reactions**, where an enzyme catalyzes a reactions (see section 1.1 for a brief explanation on

Table 1.1 Possible interpretations of MIM of figure 1.5: a blank " " means that the complex species can not be formed in some interpretation; a "✓" means that the complex species can be formed in some interpretation; a "?", in the heuristic interpretation, means that either it is not known whether those species form or that further information is provided in the text annotations attached to the map

Complex Species	**Interpretation**		
	Explicit	Heuristic	Combinatorial
A:B	✓	✓	✓
B:C	✓	✓	✓
pB	✓	✓	✓
A:C			
A:B:C		?	✓
A:pB		?	✓
pB:C		?	✓
A:pB:C		?	✓

how catalysis works), can be represented explicitly with three component reactions:

(a) binding between enzyme and substrate;

(b) dissociation of the enzyme–substrate complex;

(c) conversion of the enzyme–substrate complex to products.

Figure 1.6.(a) shows the notation using catalysis, while figure 1.6.(b) shows the same enzymatic reaction in explicit notation.

In the particular case of catalysis, Kohn *et al.* accept the use of its contingency symbol also in explicit maps. However, it will be just a compact notation for the three corresponding reactions, since, in practice, when the map is going to be simulated, the catalysis contingency will be coded into the three reactions.

Contingencies could yield some ambiguities in the meaning of a map. We can note how, depending on whether we use contingency symbols or reaction symbols to represent a certain event, it is possible to construct diverse maps having the same meaning.

In order to avoid any kind of ambiguity, Kohn *et al.* suggest not to use contingency in explicit map that must be used for computer simulation. Thus, official explicit maps have been developed using only a subset of the original symbols: reaction symbols and the catalysis symbol interpreted as reactions).

Explicit maps, without contingencies or where catalysis symbol is indeed coded into three reactions, are readily made for simulation: the set

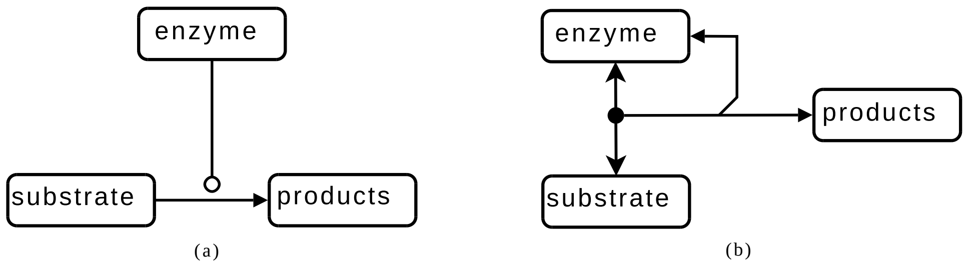

Figure 1.6 *(a)* enzymatic reaction with catalysis symbol; *(b)* enzymatic reaction in explicit notation

of possible reactions can be translated directly into input for computer simulation, yielding a set of differential equations.

Having only reactions at one's disposal, sometimes, in official maps, it could be necessary to use some reaction symbols just to depict a particular complex: this dummy reactions do not happen in practice, they serve as a visual information about the complex structure; this fact is denoted by the absence of labels on this reaction. We will see an example of this stratagem in Figure 3.9 of section 3.2.3. The same strategy is adopted for species. Species without a label are not entitled to belong to the particular system represented by the map. This stratagem is a great help in depicting maps aimed at computer simulation. In fact, in case one does not want a certain reaction depicted in the map to happen in practice, he/she can simply avoid to give this reaction a label: in this way, this reaction is not inserted it in the model constructed for the simulation. This stratagem could also help in defining and simulating sub–models of a given map: if one wants to simulate just a part of the system depicted in a map, he/she could simply labels the only reactions and species actually wanted in the simulation.

An explicit MIM, aimed at computer simulation and hence using only reaction symbols, usually comes with a molecular species table and a reaction table (also named connection table). A molecular species table describes the labeled species in the map. For each species, it is usually specified: the species label, the species identifier (that is, the extended name of the species) and a possible initial concentration. A reaction table describes the labeled reactions in the map. This table has an entry for each labeled reaction, describing reactant and product species and possibly the kinetic constant. We will see an example of this table in section 3.2.3. Looking at this table, we can clearly see which reaction is really intended to happen and which are dummy reactions. Thus, when we want to encode an explicit map in a mathematical model, we shall take into account the fact that it is possible to select a subset of reactions which is wanted to happen in practice.

Combinatorial

Beside the complexes that would be allowed by explicit interpretation, an interaction line in a MIM, in its combinatorial interpretation, represents an implicit set of complexes and hence of reactions, in particular, all those reactions between the interacting species, in each possible combination of their binding or modification state. Thus, this type of MIM defines implicitly a set of reaction paths that can take place concurrently. In Figure 1.5, the combinatorial interpretation allows binding between **A** and **B**, regardless of whether **B** is bound to **C** or phosphorylated. Kohn *et al.* calls this property "transitive", because an interaction symbol applies indirectly to species through other interaction symbols. A main advantage of the combinatorial interpretation is exactly this ability to synthesizes with few symbols a large number of possible complexes and reactions, making MIM a compact notation.

Heuristic

Like the combinatorial interpretation, the heuristic one allows all the complexes that would be allowed by explicit interpretation, with the difference that it does not specify whether each of the combinatorial possibilities may or may not occur, possibly because of lack of knowledge. Thus, heuristic MIMs serve as a compact information organizer, depicting what is known and left unspecified what still has to be discovered. This interpretation will not be of our concern, and we will not investigate further on it.

Note that, in Figure 1.5, direct binding of **A** and **C** is not allowed in any of the interpretations.

Explicit versus Combinatorial

It is interesting observing how a **cycle** of binding interactions (see Figure 1.7) is considered differently by explicit and combinatorial interpretations: the explicit interpretation allows only the formation of **A:B**, **B:C** and **C:A**, while the combinatorial one allows the formation of a chain of cyclic multimers of the form **. . . A:B:C:A:B:C . . .** , which is not a nonsense in biology. Note that, building an explicit map able to represent a possible infinite chain of cyclic multimers, would be unfeasible.

On the whole, it is possible to produce a combinatorial map with the same meaning of an explicit one, while the viceversa is not always possible, due to the possible formation of infinite chains of complexes through cycles.

We can give a combinatorial map the same meaning of an explicit one, by adding some contingency symbols, in order to prohibit those complexes

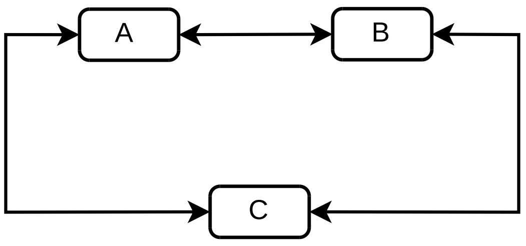

Figure 1.7 Cycle of bindings in MIM notation

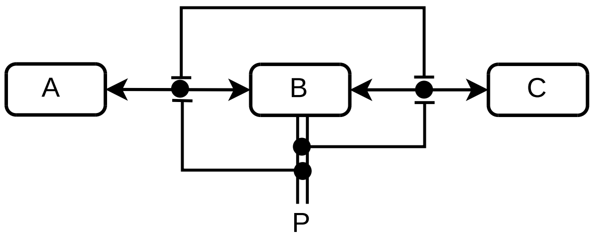

Figure 1.8 Inhibited version of the map of figure 1.5: the combinatorial interpretation of this map is equal to the explicit one of map 1.5

that an explicit map would not take into account; while, given that the map contains no cycles, we can give an explicit map the same meaning of a combinatorial one by (not so obviously) adding some reaction symbols, in order to depict all the complexes that a combinatorial interpretation would take into account.

For instance, in figure 1.8 we added few contingencies to the map of figure 1.5, in order to limit the species allowed in the combinatorial interpretation of map in figure 1.8 to those allowed in the explicit interpretation of map in figure 1.5: in particular **pB** can not bound to **A** or **C** and complex **A:B** and **B:C** are mutually exclusive. Note that we adopted a *compact notation* for the two mutual inhibitions: they are exactly the same as two inhibitions, one starting from **A:B** and pointing to **B:C** and its converse.

If a combinatorial map does not contain cycles, an explicit map can be extracted from it. As we will see in the following example, for each combinatorial map there could be several possible explicit maps and it is up to the designer do chose the most plausible one.

Let's try to build an "absurd" explicit map, which tries to be faithful to the combinatorial interpretation of the map in fig 1.5. Complex **A:B:C** can arise from the binding between **A:B** and **C** or from the binding between **A** and **B:C**. If we try to represent both of these paths, we obtain a map with the complex **A:B:C** appearing twice in the map as in Figure 1.9, thus violating one of the basic principles on which MIM relies. The same conclusion

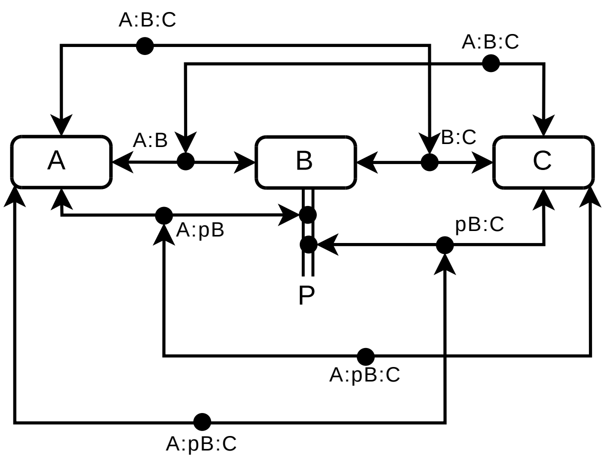

Figure 1.9 Incorrect explicit version of the combinatorial interpretation of the map in 1.5: complex species **A:B:C** and **A:pB:C** appear twice in the map

can be done for the formation of **A:pB:C**.

Since, the explicit interpretation depicts specific reaction paths, we need to choose the reaction order leading to **A:B:C** and hence choose which, among the two paths, will be represented by the explicit map. Suppose we know that **C** can only bind to **A** or **B** once these are bound together, we have that the only possible path leading to **A:B:C** is the sequence of bindings shown in the combinatorial map of Figure 1.10. The same considerations can be done in case of phosphorylated **B**. The designer should select the most reasonable path among the possible ones.

In conclusion, in constructing an explicit map with the "same" meaning of a combinatorial one, provided that there are no cycles, first we have to limit the combinatorial one to explicit paths for complexes resulting from a series of bindings, depicting these bindings in a sequential order, thus losing the concurrent peculiarity of paths in combinatorial maps. Then, we probably have to add some reaction symbols or remove superfluous contingency symbols.

As we previously noticed, explicit maps using a subset of symbols of MIM notation, are aimed at computer simulation: all the possible complexes can be listed in advance. As shown in [13, 11], it is sufficient to define a molecular species file, containing an identifying number and an initial concentration for each molecular species appearing in a map, and a reaction file, containing an entry for each reaction together with its rate constant; then, a set of ordinary differential equations (ODE) can be built from these files, in which each reaction corresponds to one term of a differential equation. The solution of a set of ODE is an ordinary and consolidated procedure nowadays. A big advantage is that modifications of constants, such as rate

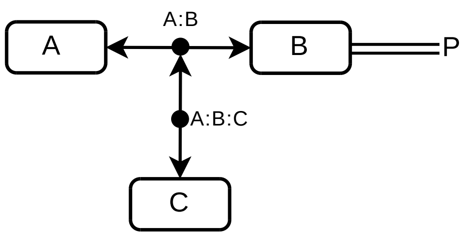

Figure 1.10 Combinatorial map depicting explicitly the order of sequential bindings yielding **A:B:C**. 1.11 is an explicit map representing the same complexes of this map.

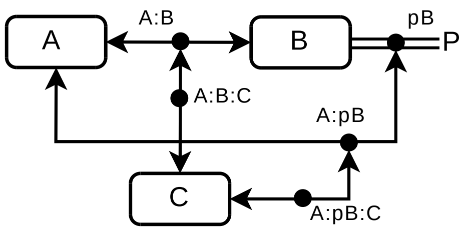

Figure 1.11 Explicit map representing the same complexes of map in Figure 1.10: complex species **A:B:C** and **A:pB:C** appear once in the map.

constants and initial concentrations, do not require the whole redefinition of the differential equations. However, modeling a map with a set of ordinary differential equations would treat as continuous aspects that are naturally discrete, such as the quantity of molecules of a given molecular species at a given time. In our formalization, we will use a stochastic rewriting language, which reflects the non–determinism of a molecular interaction map: in fact, at any time, there is a set of possible reactions that could happen in the biological system modeled with a MIM.

The combinatorial interpretation, as we have seen, is a powerful means for expressing loads of complexes with just few symbols, moreover it allows the representation of cyclic paths and it can be limited to the explicit interpretation with the addition of some contingency symbols.

The combinatorial interpretation seems more intriguing from a complexity point of view: in fact, a possibly infinite amount of different complexes can arise from a single map; moreover, the addition of single reaction line affects a whole family of complexes, namely, all those complexes directly or indirectly connected to one of the ends of the new reaction and yields to a new family of possible reactions. Therefore, a growth in the number of reactions could lead to an exponential growth in the number of possible complexes, even in a flat MIM (that is, a MIM without membranes). Our aim will be to keep the description of these complexes proportional to the size of the map; in our formalization, it will not be necessary to enumerate all possible complexes: we will define them implicitly.

From the point–of–view of simulation, a combinatorial interpretation could not be always meaningful and well understood: in fact, if a designer would like to depict a specific metabolic behavior, adopting a combinatorial interpretation could shrink the size of the map, but, on the other side, could yield meaningless or even deprecated paths. Specifying which paths should be avoided could require more work than specifying the only possible paths. From the point–of–view of science, a combinatorial interpretation could be interesting in discovering "new" paths and unexpected behaviors, even if, their biological meaning could not be clear yet.

In this work we will analyze both **explicit** and **combinatorial** interpretations of a MIM. First, we will give an extensive formal semantics for the explicit interpretation, analyzing possible reactions, contingencies and compartments created by membranes. Then, we will analyze the combinatorial interpretation, giving a possible representation for complexations and other interactions, analyzing the delicateness of translocation of complex species from a compartment to another.

1.2.5 Stochastic simulation of chemical reactions in a MIM

In this section, we recall how quantitative aspects of a map, in particular the speed and time of each reaction, can be simulated by a possible stochastic formalism, using the classical Gillespie's Algorithm [31] for stochastic simulation of chemical reactions.

In a MIM, a rate constant can be associated to each reaction symbol. If we want to simulate over time a system in a given state, we need to chose, among the possible reactions, which reaction is going to happen and when this reaction will happen. Gillespie's Algorithm for simulating chemical reactions, assumes a *reaction constant* for each considered chemical reaction, which could be derived, with some approximation, from the kinetic constant of the reaction. As said before, following the mass–action principle, the eventuality that a certain reaction happens depends on its reaction constant and on the concentrations of its reactants. Therefore, in the simulation algorithm, the probability that, at a given time, a certain reaction can happen, depends on the number of possible combination of the existing reactants multiplied by its reaction constant, we call this value *reaction rate*. For instance, the reaction

$$A + B \xrightarrow{k} C$$

in a state where we have $|A|$ molecules of kind A and $|B|$ molecules of kind B, will occur at a given time with probability depending on $\rho_k = |A||B|k$. A *state* of the simulation is the multiset of the existing molecules. Given a set of reactions $R_1, R_2, \ldots, R_n$, and ρ_i denoting the reaction rates, the *exit rate* ρ of a given state is obtained adding up each reaction rate:

$$\rho = \sum_i \rho_i$$

Given a state, a set of reactions and a value representing the current time, Gillespie's Algorithm performs two steps:

- it randomly chooses the time at which the next reaction will occur with an *exponential distribution* with parameter ρ [2];
- it randomly chooses the next reaction with probability $\frac{\rho_i}{\rho}$.

Thus, the *reaction rate* of a given reaction has a double practical use: first in determining the speed of the reaction and second to determine the probability of the reaction.

An *Exponential distribution* is a probability distribution for which holds the important *memoryless* property, by which the simulation in time can forget the history of what happened in the choice of the stochastic behavior of the next reaction. An important mathematical stochastic model, based on this exponential distribution, is the *Continuous Time Markov Chains (CTMCs).* A Continuous–Time Markov Chain is a triple (S, R, π), where S is the set of states, $R : S \times S \mapsto \Re^{\geq 0}$ is the transition function and $\pi : S \mapsto [0, 1]$ is the initial probability distribution. A CTMC satisfies the *memoryless* property: the probability of making a transition from a state to another at a given time, does not depend on the previous states. The system is assumed to pass from state s to state s' with probability $R(s, s')$, consuming an exponentially distributed quantity of time, with $R(s, s')$ as parameter of the exponential distribution; the system is assumed to start in state s with probability $\pi(s)$. If the set of state is finite, the CTMC can be represented by a square matrix with each entry representing the probability of passing from a state to another. Summing all the entries of a row corresponding to a state s, we get the exit rate for this state, which is used as parameter for an exponential distribution, to compute the time of the next transition.

Gillespie's Algorithm could be hence modeled as a CTMC model.

1.2.6 Membranes

Membranes are essential elements in a cell: they give a spatial position for elements into a cell. A species could be positioned outside a membrane, onto

[2] A random variable has an exponential distribution of parameter $\lambda \in [0, \inf]$ (also called rate of the distribution), if its probability density function as the form

$$f(X > t) = \lambda e^{-\lambda t} \qquad for \quad t \geq 0$$

a membrane, or inside a membrane. Moreover, a membrane could contain other membranes, thus forming a nested structure.

A diagram built in MIM notation is "flat": it does not provide a means to describe spatial structure. Although MIM notation does not provide a formal notation to depict membranes in a diagram, in practice membranes have been sketched in some MIM diagrams and species have been placed with respect of this membrane structure (see Figure 11 in [10]).

A MIM diagram does not state any spatial limitation in depicting interactions due to membranes: in principle, it is correct to connect a species inside a membrane to another one outside the same membrane, even if the interaction line could possibly traverse a series of membranes; it will be up to the designer to place the respective product in the correct membrane.

Membranes could yield different interpretations of interactions: we could assume that an interaction between two species can happen only in the particular place in the map where this interaction is depicted (local interactions) or we could assume that the interaction can happen at any place, that is independently of the particular place where the interaction line is depicted, still holding all the other conditions for the interaction to happen (global interactions).

Just to keep the previous terminology, we could call the first local interpretation "explicit", since a single interaction line represents a single possible interaction, that is the one happening in that exact place where the line is depicted, while, the second global interpretation could be seen as "combinatorial", since a single interaction line would implicitly represent a set of interactions, one interaction for each possible place where the given interaction could happen.

In the present work, we retain the first interpretation. The main reason is that assuming that an interaction can happen only in the exact place in the map where it is depicted, looks reasonable from a biological point–of–view, since different membranes could have a different state, such as different Ph, temperature and all those particular conditions which could make an otherwise impossible interaction possible.

As far as we know, MIM notation does not provide means for describing membrane breaking, joining or other operations on membranes, thus we will assume the membrane structure to be statical over time.

1.2.7 Example of MIM diagram

Figure 1.12 shows a MIM diagram with reaction and contingency symbols, membranes and some multidomain species. This example could make no sense from the biological point–of–view: it is only intended to show some MIM notation and how a map could be put together with the symbols provided by the notation, thus, in this context, it is not of our concern specifying

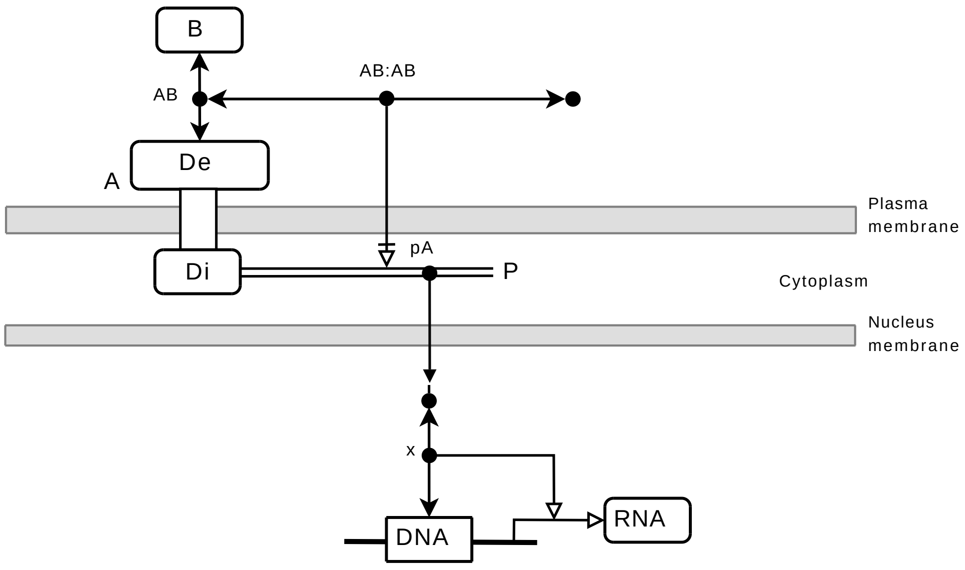

Figure 1.12 Example of MIM diagram (see text for explanation)

whether the map should be interpreted explicitly or combinatorially. The diagram represents a cell, delimited by its plasma membrane and containing its nucleus, which is delimited by another inner membrane. Lying on the plasma membrane, there is a receptor protein **A**, which is represented as a multi–domain molecular species: it has an external domain **De** and an internal domain **Di**. The external domain could bind to a molecule **B** outside the cell, let call **AB** this compound. Two different instances of the **AB** compound can eventually bind together (this is represented with the “isolated–node” convention), forming a dimer **AB:AB**. This dimer is necessary for some internal domain of the molecular species **A** to be phosphorylated, yielding **pA**. The phosphorylated molecule can eventually move from the cytoplasm to the nucleus and bind to the **DNA** stimulating the transcription of some genes.

In this example it is clear that the isolated–node symbol needs to be disambiguated. In particular, it is not clear if **x** represents the dimer formed by two **DNA** instances, to which **pA** stoichiometrically converts or if **x** represents **pA** translocated to the nucleus.
In order to disambiguate this situation, a short line, pointing to the species represented by the isolated–node, is used.

Note that this time we indicate the complex **A:B** with **AB**. Indeed, it makes no difference from a semantical point–of–view, they are just two possible names for a complex molecular species, we could have also named it **C**. It could be recommended to use names which carry information about complex formation, but when the "colon" notation could become clumsy, the name can of course be shrunk as preferred (in this case, we wanted to avoid names like **(A:B):(A:B)**, which is quite long, or even worse **A:B:A:B**, which could be misleading.

We would like to remark how, according to the kind of interpretation we give to the map, its meaning can be completely twisted. For instance, in its explicit interpretation, dimer **AB:AB** is necessary for a molecule **A**, in its elementary state, to be phosphorylated; while, in its combinatorial interpretation, dimer **AB:AB** is necessary for **A**, in any of its binding state, to be phosphorylated. Therefore, using the usual *colon ":" notation* for complexes, we could say that:

- in the explicit interpretation, the only possible phosphorylated molecule is **pA**;
- in the combinatorial interpretation, we could have **pA**, **pAB**, **p(AB:AB)**;

Thus, in the combinatorial interpretation, we can see dimer **AB:AB** as necessary for itself (the exact same molecule) being phosphorylated[3]. In practice, this would reasonably mean that, before **AB:AB** can be phosphorylated, **AB:AB** must exist. This simple condition, which establishes an order between possible reactions, could be described by an *explicit* stoichiometric conversion of **AB:AB** into **p(AB:AB)**. With this example, we just wanted to stress how, depicting desired complexes using a combinatorial map, could be tricky.

[3]Note that, if we wanted to express that **AB:AB** is necessary for the phosphorylation of another molecule of species **AB:AB**, we could have used the *in–trans* symbol

Chapter 2

Calculi of Looping Sequences

2.1 A Formalism for Biological Systems

In our task of giving a formal semantics for MIMs, we need an appropriate formalism for describing the biological systems of our concern, which shall be able to describe the components and interactions of interest in a molecular interaction map. In particular, this formalism should allow the description of elementary and complex molecular species and the possible interactions among them, possibly also at domain/site level, such as reactions between two different domains or modification of a molecule at a particular site. Moreover, the formalism should be able to describe the physical structure of the system represented in a MIM, taking into account the possible hierarchical structure formed by membranes and the spatial positioning of species in terms of this membrane structure.

To our purposes, we use formalisms in that family of calculi termed *Calculi of Looping Sequences*, developed by Barbuti R., Maggiolo–Schettini A., Milazzo P. and Troina A. at the University of Pisa. This family of calculi includes several formalisms: Full–CLS, CLS, CLS+, LCLS, Stochastic CLS. In the present work, Full–CLS is not of main concern and it will not be presented. For an extensive reading about the definition of the formalisms, their differences and some application see [29].

The *Calculus of Looping Sequences* (CLS, [27]) is a calculus based on term rewriting, suitable to the description of biological systems. It allows the description of proteins, DNA and RNA fragments, membranes and all other macromolecules, keeping track of the physical structure of the described system and having a fairly simple syntax and semantics. In CLS, elements appearing onto a membrane are modeled as a single "chain" of elements, connected in a sequence.

CLS+ (*CLS plus*, [27]) is an extension of CLS, which allows parallel elements (modeled has parallel sequences) to appear onto a membrane.

Stochastic CLS [28] is an extension of CLS, which adds the ability of modeling quantitative aspects of biological systems, such as time and probability of having a certain evolution of the system.

LCLS (*Calculus of Linked Looping Sequences*, [33, 30]:unpublished) is another extension of CLS, which adds the ability of modeling molecular interactions at domain/site level. Notice that, to our purposes, the interacting abilities of molecular sites or molecular domains are equivalent: therefore, in our formal specification a domain or site will be represented by the same formal symbol.

As previously mentioned, in our work we will cover formally and extensively the aspects of the **explicit** interpretation of a MIM and in a less formal way some of the aspects of the **combinatorial** one. For each interpretation, we chose a calculus of looping sequences that could suit best the its peculiarity.

Concerning explicit maps, we chose a stochastic extension of CLS+. Indeed, CLS could have worked as fine as CLS+, in giving a formalization of explicit MIMs. However, the *plus* extension simplifies our work, since, as we will see, we only have to deal with parallel compositions of species, whether the species are situated onto a membrane or inside a membrane (we remark that CLS does not allow commutativity on membrane surfaces and a set of species onto a membrane would be represented as a single sequence of concatenated species, while in CLS+ it would be a parallel composition of sequences).

Concerning combinatorial maps, we chose a stochastic extension of LCLS+ (that is, LCLS hypothetically *Plus* extended). The ability of modeling interactions at domain level is very useful in describing a combinatorial interpretation of complexes: if we can name each domain or interaction site univocally in the map, and we assume that a reaction line in a MIM connects two disjoint domains, we can identify a reaction with the pair of the corresponding interacting domains; this will be the main idea behind our possible formalization of combinatorial maps. As we will see, this representation of reactions, as links between a unique pair of domains, requires some effort in formalizing the translocation from a membrane to another of a linked complex. On the other side, the formalization of "flat" combinatorial maps, which do not have a membrane structure, is less tricky. In this chapter, we give a short introduction to LCLS. A more comprehensive semantics of LCLS can be found in appendix A.

In both explicit and combinatorial cases, we adopted a possible stochastic extension of the chosen calculus: we say "possible" because a formal stochastic extension has been given only for CLS. Although there is not a formal description of this hypothetical calculi, we think that their semantics should not differ too much from the one given for Stochastic CLS.

In the next sections we recall the Calculus of Looping Sequences and its extensions. Throughout these sections, we try to trace out an idea on how the calculus could represent biological aspects of our concern, in particular we report the modeling guidelines given in [27] for modeling biomolecular events in CLS. This section should be of great interest for the reader who wants to have an anticipation on how a biomolecular event could be modeled in practice in CLS by a term and some rewrite rule.

2.2 The Calculus of Looping Sequences

The Calculus of Looping Sequences is a formalism based on term rewriting. A model defined in CLS consists of a *term* and a *set of rewrite rule.* A term represents the actual configuration of the modeled system, while the set of rewrite rules describes the possible events and how the system could possibly evolve.

A term could be a sequence, composed by zero or more symbols in a given alphabet, a looping sequence containing a term, or a parallel composition of two terms. A looping sequence represents a sequence, whose end symbols are connected in circle. The "looping" attribute underlines the rotation property of the elements of such a sequence, as we shall see in the following. The formal syntax for terms is given by the following grammar, where we assume a possibly infinite alphabet $\mathcal{E}$ of symbols ranged over by $a, b, c, \ldots$.

Definition 2.1 (Terms)**.** Terms T *and* Sequences S *of CLS are given by the following grammar:*

$$\begin{array}{rcl} T & ::= & S \quad \big| \quad (S)^L \rfloor T \quad \big| \quad T \,|\, T \\ S & ::= & \epsilon \quad \big| \quad a \quad \big| \quad S \cdot S \end{array}$$

where a is a generic element of $\mathcal{E}$ and ϵ represents the empty sequence (a sequence of zero symbols). The infinite set of terms is denoted by $\mathcal{T}$, and the infinite set of sequences by $\mathcal{S}$.

CLS syntax provides us with three operators:

Sequencing $\_ \cdot \_$ a binary operator, which takes two sequences and concatenates them, creating a new sequence;

Looping–and–Containment $(\_)^L \rfloor \_$ a binary operator, which takes a sequence and a term and builds a looping sequence containing the given term;

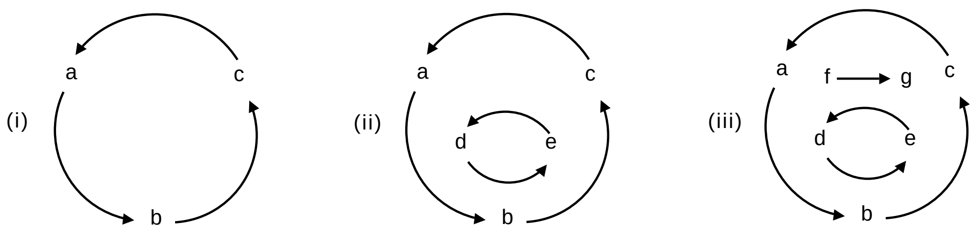

Figure 2.1 (i) represents $\big(a \cdot b \cdot c\big)^L \rfloor \epsilon$; (ii) represents $\big(a \cdot b \cdot c\big)^L \rfloor \big(d \cdot e\big)^L \rfloor \epsilon$; (iii) represents $\big(a \cdot b \cdot c\big)^L \rfloor \big(\big(d \cdot e\big)^L \rfloor \epsilon \,|\, f \cdot g\big)$.

Parallel composition $\_ \,|\, \_$ a binary parallel composition operator, which takes two terms and juxtaposes them.

Brackets can be used to indicate the order of application of the operators, and we assume $\big(\_\big)^L \rfloor \_$ to have the precedence over $\_ \,|\, \_$, therefore $\big(a\big)^L \rfloor d \cdot e \,|\, f$ stands for $\big(a\big)^L \rfloor (d \cdot e) \,|\, f$. Figure 2.1 shows some examples of CLS terms and their visual representation.

Several biological components could be easily abstracted as a sequence: for instance, a DNA strand can be seen as a sequence of nucleic acids or, at higher level, a sequence of genes; a protein can be seen as a sequence of amino acids or a sequence of interaction sites. In general, each multi–domain or multi–site molecular species could be represented as a sequence of basic elements, each element standing for a single site or domain.

A membrane can be seen as a closed surface, which can be interspersed with molecules and can contain something, which could also be other membranes. Thus, a membrane could be abstracted by a looping sequence of the elements appearing onto the membrane, containing what is inside the membrane itself. The nested structure formed by membranes, is naturally abstracted by nesting of looping sequences. Note that the syntax of terms imposes that the looping operator can be applied only to a sequence and hence, objects lying onto a membrane are represented by subsequences of this looping sequence. This limitation leads to a less natural representation of membranes: we are used to think about membranes as closed barriers on which particles are disjoint one from each other and can move freely. On one side, this syntactical restriction will allow the definition of a simple structural congruence relation on sequences on terms and avoids some ambiguities in the meaning of terms, but, on the other side, the commutative property of objects on a membrane is lost.

Definition 2.2 (Structural Congruence)**.** *The structural congruence relations $\equiv_S$ and $\equiv_T$ are the least congruence relations on sequences and on* *terms, respectively, satisfying the following rules:*

$A1.\quad S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3$

$A2.\quad S \cdot \epsilon \equiv_S \epsilon \cdot S \equiv_S S$

$A3.\quad S_1 \equiv_S S_2$ *implies* $S_1 \equiv_T S_2$ *and* $\big(S_1\big)^L \rfloor T \equiv_T \big(S_2\big)^L \rfloor T$

$A4.\quad T_1 \,|\, T_2 \equiv_T T_2 \,|\, T_1$

$A5.\quad T_1 \,|\, (T_2 \,|\, T_3) \equiv_T (T_1 \,|\, T_2) \,|\, T_3$

$A6.\quad T \,|\, \epsilon \equiv_T T$

$A7.\quad \big(\epsilon\big)^L \rfloor \epsilon \equiv \epsilon$

$A8.\quad \big(S_1 \cdot S_2\big)^L \rfloor T \equiv_T \big(S_2 \cdot S_1\big)^L \rfloor T$

Axiom $A1$ states the associativity of _ · _; axiom $A3$ propagates congruence of sequences into congruence of looping sequences; axiom $A4$ and $A5$ state the commutativity and associativity of _ | _, respectively; axioms $A2$, $A6$ and $A7$ state the neutral role of ϵ with respect to the operators of the calculus; axiom $A8$ motivates the "looping" attribute and states that sequences enclosed in a loop can rotate. In the following, for the sake of simplicity, $\equiv$ will be used in place of $\equiv_T$.

Once we have a term, describing the actual structure and components of the modeled system, we need to define a set of rewrite rules to describe how this components and structure evolve in time.

A rewrite rule is a pair of terms (T_1, T_2), describing how a term changes when a particular event occur: T_1 describes the portion of the system where the rule can be applied and T_2 describes how that portion of the system evolves when the event occurs.

In order to increase the expressiveness of a rewrite rule, the calculus allows the use of variables in the rule definition: therefore, a rule can be applied to all terms, which can be obtained by properly instantiating its variables. In CLS, there are three type of variables: term, sequence and single alphabet element variables, which belongs respectively to one of the following infinite and pairwise disjoint sets of variables:

Term variables TV**,** ranged over by $X, Y, Z, \ldots$;

Sequence variables SV**,** ranged over by $\widetilde{x}, \widetilde{y}, \widetilde{z}, \ldots$;

Element variables $\mathcal{X}$**,** ranged over by $x, y, z, \ldots$.

The calculus denotes the set of all variables by $\mathcal{V} = TV \cup SV \cup \mathcal{X}$ and a generic variable of $\mathcal{V}$ by ρ.

Patterns are terms with the addition of variables. A pattern containing no variables is simply a term, and it is called *ground pattern*. The following definition describes the syntax of patterns.

Definition 2.3 (Patterns)**.** *Patterns P and* Sequence Patterns *SP of CLS are given by the following grammar:*

$$\begin{array}{rcl} P & ::= & SP \;\big|\; \big(SP\big)^L \rfloor P \;\big|\; P \,|\, P \;\big|\; X \\ SP & ::= & \epsilon \;\big|\; a \;\big|\; SP \cdot SP \;\big|\; \widetilde{x} \;\big|\; x \end{array}$$

where a is a generic element of $\mathcal{E}$, and $X, \widetilde{x}$ and x are generic elements of TV, SV and $\mathcal{X}$, respectively. The infinite set of patterns is denoted by $\mathcal{P}$.

The structural congruence relation is assumed to be trivially extended to patterns.

An *instantiation* is a partial function $\sigma : \mathcal{V} \to \mathcal{T}$. An instantiation must preserve the type of variables, thus for $X \in TV, \widetilde{x} \in SV$ and $x \in \mathcal{X}$ we have $\sigma(X) \in \mathcal{T}, \sigma(\widetilde{x}) \in \mathcal{S}$ and $\sigma(x) \in \mathcal{E}$, respectively. Given $P \in \mathcal{P}$, $P\sigma$ denotes the ground term obtained by replacing each occurrence of each variable $X \in \mathcal{V}$ appearing in P with the corresponding term $\sigma(X)$. Σ denotes the set of all the possible instantiations and, given $P \in \mathcal{P}$, $Var(P)$ denotes the set of variables appearing in P.

The following is the definition of *rewrite rule* in CLS.

Definition 2.4 (Rewrite Rules)**.** *A rewrite rule is a pair of patterns (P_1, P_2), denoted with $P_1 \mapsto P_2$, where $P_1, P_2 \in \mathcal{P}$, $P_1 \not\equiv \epsilon$ and such that $Var(P_2) \subseteq Var(P_1)$. The infinite set of all the possible rewrite rules is denoted by $\Re$. A rewrite rule is* ground *if $Var(P_1) = Var(P_2) = \emptyset$, and a set of rewrite rules $\mathcal{R} \subseteq \Re$ is* ground *if all the rewrite rules it contains are ground.*

A rewrite rule (P_1, P_2) states that a term $P_1\sigma$, obtained by instantiating variables in P_1 by some instantiation function σ, can be transformed into the term $P_2\sigma$.

The *semantics* of CLS is defined as a transition system, in which states correspond to terms and transitions correspond to rule applications.

Definition 2.5 (Semantics)**.** *Given a set of rewrite rules $\mathcal{R} \subseteq \Re$, the* semantics *of CLS is the least transition relation $\to$ on terms closed under $\equiv$, and satisfying the following inference rules:*

$$\frac{(P_1, P_2) \in \mathcal{R} \qquad P_1\sigma \not\equiv \epsilon \qquad \sigma \in \Sigma}{P_1\sigma \to P_2\sigma}$$

$$\frac{T_1 \to T_2}{T \,|\, T_1 \to T \,|\, T_2} \qquad \frac{T_1 \to T_2}{\big(S\big)^L \rfloor T_1 \to \big(S\big)^L \rfloor T_2}$$

where the symmetric rule for the parallel composition is omitted.

A CLS *model* is given by a term, describing the initial state of the modeled system and a set of rewrite rules, describing all the possible events which makes the term evolve.

Biomolecular Entity	**CLS Term**
Elementary object (genes, domains, other molecules, etc...)	Alphabet symbol
DNA strand	Sequence of elements representing genes
RNA strand	Sequence of elements representing transcribed genes
Protein	Sequence of elements representing domains or single alphabet symbol
Molecular population	Parallel composition of molecules
Membrane	Looping sequence

Table 2.1 Guidelines for the abstraction of biomolecular entities into CLS.

2.2.1 CLS Modeling Guidelines

In this section we report the general guidelines for modeling biomolecular systems in CLS as given in [27].

Table 2.1 describes how CLS terms could be associated to biomolecular entities, such as proteins, membranes and a population of molecules. We already anticipated some of this guidelines in the previous section. Elementary objects are modeled as alphabet symbols, for instance DNA could represent a molecule of **DNA**, A could represent a molecule of species **A**. Non–elementary objects could be represented by CLS sequences. However, we would like to notice that a complex species could also be modeled as an alphabet symbol, for instance the complex formed by the binding of a molecule A to a molecule B, could be described with C or, in a more informational manner, with $A\!:\!B$. CLS sequences are suitable to represent molecule at domain level, and, in general, they can give some more information about the inner structure of a molecule, for instance, a protein could be represented by the sequence of its amino–acids when needed. CLS sequences become of particular interest when we want to model possible interactions at domain level: as we shall see, such interactions can not be properly modeled in CLS, for this reason its linked extension has been proposed. Membranes are modeled by looping sequences.

Table 2.2 describes some examples of how CLS rewrite rules could be associated to biomolecular events, such as complexation, decomplexation, catalysis, membrane joining, membrane fusion, etc. We already remarked that, in the present work, we assume the membrane structure to be statical, thus, we will not need to represent events on membrane. We can anyway notice how these membrane events could be easily modeled in CLS.

We would like the reader to note how a complexation could be ex-

Biomolecular Event	**Examples of CLS Rewrite Rule**
State change	$a \mapsto b$ $\widetilde{x} \cdot a \cdot \widetilde{y} \mapsto \widetilde{x} \cdot b \cdot \widetilde{y}$
Complexation	$a \mid b \mapsto c$ $\widetilde{x} \cdot a \cdot \widetilde{y} \mid b \mapsto \widetilde{x} \cdot c \cdot \widetilde{y}$
Decomplexation	$c \mapsto a \mid b$ $\widetilde{x} \cdot c \cdot \widetilde{y} \mapsto \widetilde{x} \cdot a \cdot \widetilde{y} \mid b$
Catalysis	$c \mid P_1 \mapsto c \mid P_2$ where $P_1 \mapsto P_2$ is the catalyzed event
State change on membrane	$\big(a \cdot \widetilde{x}\big)^L \rfloor X \mapsto \big(b \cdot \widetilde{x}\big)^L \rfloor X$
Complexation on membrane	$\big(a \cdot \widetilde{x} \cdot b \cdot \widetilde{y}\big)^L \rfloor X \mapsto \big(c \cdot \widetilde{x} \cdot \widetilde{y}\big)^L \rfloor X$ $a \mid \big(b \cdot \widetilde{x}\big)^L \rfloor X \mapsto \big(c \cdot \widetilde{x}\big)^L \rfloor X$ $\big(b \cdot \widetilde{x}\big)^L \rfloor (a \mid X) \mapsto \big(c \cdot \widetilde{x}\big)^L \rfloor X$
Decomplexation on membrane	$\big(c \cdot \widetilde{x}\big)^L \rfloor X \mapsto \big(a \cdot b \cdot \widetilde{x}\big)^L \rfloor X$ $\big(c \cdot \widetilde{x}\big)^L \rfloor X \mapsto a \mid \big(b \cdot \widetilde{x}\big)^L \rfloor X$ $\big(c \cdot \widetilde{x}\big)^L \rfloor X \mapsto \big(b \cdot \widetilde{x}\big)^L \rfloor (a \mid X)$
Catalysis on membrane	$\big(c \cdot \widetilde{x} \cdot SP_1 \cdot \widetilde{y}\big)^L \mapsto \big(c \cdot \widetilde{x} \cdot SP_2 \cdot \widetilde{y}\big)^L$ where $SP_1 \mapsto SP_2$ is the catalyzed event
Membrane crossing	$a \mid \big(\widetilde{x}\big)^L \rfloor X \mapsto \big(\widetilde{x}\big)^L \rfloor (a \mid X)$ $\big(\widetilde{x}\big)^L \rfloor (a \mid X) \mapsto a \mid \big(\widetilde{x}\big)^L \rfloor X$ $\widetilde{x} \cdot a \cdot \widetilde{y} \mid \big(\widetilde{z}\big)^L \rfloor X \mapsto \big(\widetilde{z}\big)^L \rfloor (\widetilde{x} \cdot a \cdot \widetilde{y} \mid X)$ $\big(\widetilde{z}\big)^L \rfloor (\widetilde{x} \cdot a \cdot \widetilde{y} \mid X) \mapsto \widetilde{x} \cdot a \cdot \widetilde{y} \mid \big(\widetilde{z}\big)^L \rfloor X$
Catalyzed membrane crossing	$a \mid \big(b \cdot \widetilde{x}\big)^L \rfloor X \mapsto \big(b \cdot \widetilde{x}\big)^L \rfloor (a \mid X)$ $\big(b \cdot \widetilde{x}\big)^L \rfloor (a \mid X) \mapsto a \mid \big(b \cdot \widetilde{x}\big)^L \rfloor X$ $\widetilde{x} \cdot a \cdot \widetilde{y} \mid \big(b \cdot \widetilde{z}\big)^L \rfloor X \mapsto \big(b \cdot \widetilde{z}\big)^L \rfloor (\widetilde{x} \cdot a \cdot \widetilde{y} \mid X)$ $\big(b \cdot \widetilde{z}\big)^L \rfloor (\widetilde{x} \cdot a \cdot \widetilde{y} \mid X) \mapsto \widetilde{x} \cdot a \cdot \widetilde{y} \mid \big(b \cdot \widetilde{z}\big)^L \rfloor X$
Membrane joining	$\big(\widetilde{x}\big)^L \rfloor (a \mid X) \mapsto \big(a \cdot \widetilde{x}\big)^L \rfloor X$ $\big(\widetilde{x}\big)^L \rfloor (\widetilde{y} \cdot a \cdot \widetilde{z} \mid X) \mapsto \big(\widetilde{y} \cdot a \cdot \widetilde{z} \cdot \widetilde{x}\big)^L \rfloor X$
Catalyzed membrane joining	$\big(b \cdot \widetilde{x}\big)^L \rfloor (a \mid X) \mapsto \big(a \cdot b \cdot \widetilde{x}\big)^L \rfloor X$ $\big(\widetilde{x}\big)^L \rfloor (a \mid b \mid X) \mapsto \big(a \cdot \widetilde{x}\big)^L \rfloor (b \mid X)$ $\big(b \cdot \widetilde{x}\big)^L \rfloor (\widetilde{y} \cdot a \cdot \widetilde{z} \mid X) \mapsto \big(\widetilde{y} \cdot a \cdot \widetilde{z} \cdot \widetilde{x}\big)^L \rfloor X$ $\big(\widetilde{x}\big)^L \rfloor (\widetilde{y} \cdot a \cdot \widetilde{z} \mid b \mid X) \mapsto \big(\widetilde{y} \cdot a \cdot \widetilde{z} \cdot \widetilde{x}\big)^L \rfloor (b \mid X$
Membrane fusion	$\big(\widetilde{x}\big)^L \rfloor (X) \mid \big(\widetilde{y}\big)^L \rfloor (Y) \mapsto \big(\widetilde{x} \cdot \widetilde{y}\big)^L \rfloor (X \mid Y)$
Catalyzed membrane fusion	$\big(a \cdot \widetilde{x}\big)^L \rfloor (X) \mid \big(b \cdot \widetilde{y}\big)^L \rfloor (Y) \mapsto$ $\big(a \cdot \widetilde{x} \cdot b \cdot \widetilde{y}\big)^L \rfloor (X \mid Y)$
Membrane division	$\big(\widetilde{x} \cdot \widetilde{y}\big)^L \rfloor (X \mid Y) \mapsto \big(\widetilde{x}\big)^L \rfloor (X) \mid \big(\widetilde{y}\big)^L \rfloor (Y)$
Catalyzed membrane division	$\big(a \cdot \widetilde{x} \cdot b \cdot \widetilde{y}\big)^L \rfloor (X \mid Y) \mapsto$ $\big(a \cdot \widetilde{x}\big)^L \rfloor (X) \mid \big(b \cdot \widetilde{y}\big)^L \rfloor (Y)$

Table 2.2 Guidelines for the abstraction of biomolecular events into CLS.

pressed in CLS. The first rule describes that a molecule a can bind to molecule b to form the complex named c. The second rule replicates the same binding in the context in which a is a sequence element, that could be the representation of a binding at domain level: in this case, a is an element of any sequence and could represent a domain of a certain molecule (note how variables can be helpful in describing a set of terms), b could be the unique domain of another molecule, c is the complex given by binding of domain a to domain b. Note how, in CLS, we are suggested to place the complex c in the same position where a appeared in the sequence. This could lead to a loss of information about the domains of the original reactant molecules and. We here anticipate the problem of modeling interactions at domain level with CLS: as we will see in the following, the linked extension of CLS addresses this problem. We will make some further example to clarify this situation in the next subsection.

We remark that we will be interested in representing domains of molecular species as sequences only when we will analyze a possible formalization for combinatorial maps: in the explicit interpretation, as we will see, the first rule for complexation, given in the table, is enough for representing complexes and also intra–domain bindings. In fact, in this case, a molecule with an intra–domain bond, will be simply a new molecular species.

2.2.2 Bindings in CLS

In this section, we try to show why, in case we need to represent interactions at domain level, CLS will not be expressive enough. As we will see, CLS can be enough to represent all kind of interactions in case of explicit maps, since in this case, we can list in advance all the possible complexes, that could arise from the map and we can just give an univocal name to each possible complex, whether this results from bindings at domain level or not. As we have seen, listing all possible complexes in advance is generally not possible for a combinatorial map: in this case, we will need to explicitly model information about domains and interactions between domains; in particular, a molecular species with some domain will be represented by a sequence containing its domains as alphabet elements.

As we have seen, In CLS, in case we are not interested to what happens in a molecule at the domain level, we could easily describe a binding and an eventual unbinding between two molecules A and B respectively, with the following rewrite rules:

$$A \mid B \;\mapsto\; A{:}B$$

$$A{:}B \;\mapsto\; A \mid B$$

The first rule states that binding of A and B gives rise to a freshly new compound, namely $A{:}B$. This interpretation of binding is quite in accordance

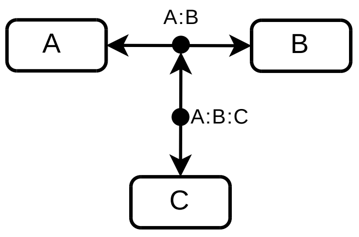

Figure 2.2 A sequence of bindings in MIM notation: A must bind to B, before the resulting compound can bind to C

with what happens in reality. In the possibility that $A{:}B$ could then bind to another molecule C, we would add the following rules:

$$A{:}B \mid C \;\mapsto\; A{:}B{:}C$$

$$A{:}B{:}C \;\mapsto\; A{:}B \mid C$$

Note that this set of four rewrite rules corresponds to the sequential bindings in MIM notation of Figure 2.2.

In analyzing a possible formal semantics for combinatorial maps, we would like to keep all the information about domains and their interaction capabilities and we would like an effective way of representing them.

As we previously said, multi–domain species could be represented in CLS as a sequence of basic alphabet elements, each element standing for a single domain. We would like to be able to express both intra–molecular bindings, that is linking between two different domains belonging to the same molecule, and inter–molecular bindings, that is linking between two domains belonging to two different molecules.

Let's think about how to represent such bindings In CLS. If, similarly to what we did in the previous example, we try to figure out a new compound as a result of the binding, which matches our aim of *keeping track of the interaction capabilities of each domain*, we could end up with the following "solution": we could, for instance, think about describing an intra–molecular binding between domains a and b of a same molecule, using the following rewrite rule:

$$a \cdot \widetilde{x} \cdot b \cdot \widetilde{y} \;\mapsto\; ab \cdot \widetilde{x} \cdot \widetilde{y}$$

where, domains a and b "disappear" from the initial molecule, and a new domain ab appears. For instance, we could apply this rule to the sequence $a \cdot c \cdot c \cdot b \cdot d$, obtaining the following transition:

$$a \cdot c \cdot c \cdot b \cdot d \;\mapsto\; ab \cdot c \cdot c \cdot d$$

It is clear that we will have an immediate problem in reversing the binding:

$$ab \cdot \widetilde{x} \;\mapsto\; a \cdot b \cdot \widetilde{x}$$

The unbinding of ab, does not preserve the original order of a and b.

Trying to describe inter–domains bindings in CLS, arises even more noticeable problems. For instance, let's have two molecules $a \cdot b$ and $c \cdot d$, how do we represent the fact that domain a can bind to domain c? We could express the result as the concatenation of the two sequences, with an auxiliary unique symbol that keeps track of the point of division of the new compound, as expressed by the following rules:

$$a \cdot \widetilde{x} \,|\, c \cdot \widetilde{y} \;\mapsto\; a \cdot \widetilde{x} \cdot \mathit{break} \cdot c \cdot \widetilde{y}$$

$$\widetilde{x} \cdot \mathit{break} \cdot \widetilde{y} \;\mapsto\; \widetilde{x} \cdot \widetilde{y}$$

The binding of our two terms would result in $a \cdot b \cdot \mathit{break} \cdot c \cdot d$, but this poor solution would miserably ignore the existence of domains, since it would not express which domain binds to which other. A possible solution, would be to mark the interacting domains with the same label, meaning that they are bound to each other, for instance, we could use the following rules:

$$a \cdot \widetilde{x} \,|\, c \cdot \widetilde{y} \;\mapsto\; a \cdot l \cdot \widetilde{x} \,|\, c \cdot l \cdot \widetilde{y}$$

$$x \cdot l \cdot \widetilde{x} \,|\, y \cdot l \cdot \widetilde{y} \;\mapsto\; x \cdot \widetilde{x} \,|\, y \cdot \widetilde{y}$$

In general, marking a symbol with the syntax of CLS, complicates the definition of a semantics for terms, which has to take into account all this "syntactical tricks", due to a notation not expressive enough.

The linked extension of CLS (LCLS), which will be presented in a future section, addresses this problem.

2.3 CLS+

We briefly describe the *plus* extension of CLS, which allows the looping operator to be applied to a parallel composition of sequences. The price for a more natural way of representing membranes, will be a slightly more complex semantics. For a comprehensive description of this calculus, please refer to [29].

The following is the definition of Terms in CLS+.

Definition 2.6 (Terms)**.** *Terms T, Branes B, and Sequences S of CLS+ are given by the following grammar:*

$$\begin{array}{lcl} T & ::= & S \;\big|\; (B)^L \rfloor T \;\big|\; T \,|\, T \\ B & ::= & S \;\big|\; S \,|\, S \\ S & ::= & \epsilon \;\big|\; a \;\big|\; S \cdot S \end{array}$$

where a is a generic element of $\mathcal{E}$. The infinite set of terms is denoted by $\mathcal{T}$, the infinite set of branes is denoted by $\mathcal{B}$ and with the infinite set of sequences is denoted by $\mathcal{S}$.

Branes forms a new syntactical category in CLS+. This requires the addition of specific formalisms for their manipulation, such as a proper Structural Congruence Relation, Brane Variables, a proper Semantic Rule, etc.

The interesting aspect of the new Structural Congruence Relation is that the ability of looping sequences to rotate is replaced by commutativity of branes. This allows an element on a membrane to freely move on it. The structural congruence rule for rotation is hence removed, while a specific structural congruence relation on branes, $\equiv_B$ is given in addition to those given for CLS. The new structural congruence relation is defined as following:

Definition 2.7 (Structural Congruence)**.** *The structural congruence relations $\equiv_S$, $\equiv_B$ and $\equiv_T$ are the least congruence relations on sequences, on branes and on terms, respectively, satisfying the following rules:*

$$S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3 \qquad S \cdot \epsilon \equiv_S \epsilon \cdot S \equiv_S S$$

$$S_1 \equiv_S S_2 \textit{ implies } S_1 \equiv_B S_2$$
$$B_1 \,|\, B_2 \equiv_B B_2 \,|\, B_1 \qquad B_1 \,|\, (B_2 \,|\, B_3) \equiv_B (B_1 \,|\, B_2) \,|\, B_3 \qquad B \,|\, \epsilon \equiv_B B$$

$$S_1 \equiv_S S_2 \textit{ implies } S_1 \equiv_T S_2$$
$$B_1 \equiv_B B_2 \textit{ implies } \big(B_1\big)^L \rfloor T \equiv_T \big(B_2\big)^L \rfloor T$$
$$T_1 \,|\, T_2 \equiv_T T_2 \,|\, T_1 \qquad T_1 \,|\, (T_2 \,|\, T_3) \equiv_T (T_1 \,|\, T_2) \,|\, T_3 \qquad T \,|\, \epsilon \equiv_T T \qquad \big(\epsilon\big)^L \rfloor \epsilon \equiv \epsilon$$

In the definition of patterns, a new variable type for Brane has been added: namely BV ranged over by $\overline{x}, \overline{y}, \overline{z}, \ldots$.

Rewrite rules are still pairs of patterns. The novelty respect CLS, is that now it will be possible to apply a rule like $a \,|\, b \mapsto c$ to elements of a looping sequence. For instance the term $\big(a \,|\, b\big)^L \rfloor d$ could be rewritten into $\big(c\big)^L \rfloor d$.

To take into account this kind of rules, the subset of Brane Rules $\Re_{\mathcal{B}} \subset \Re$ is defined has the set of rules having the form (B_1, B_2) with $B_1, B_2 \in \mathcal{B}$. The semantics of CLS+ will allow these brane rules to be applied only to Brane elements, through the addition of a specific transition relation $\rightarrow_{\mathcal{B}}$ on branes. Just as in CLS, a CLS+ model will be composed by a term and a set of rewrite rules.

The following is the formal definition for the semantics of CLS+.

Definition 2.8 (Semantics)**.** *Given a set of rewrite rules $\mathcal{R} \subseteq \Re$, and a set of brane rules $\mathcal{R}_{\mathcal{B}} \subseteq \mathcal{R}$, such that $(\mathcal{R} \setminus \mathcal{R}_{\mathcal{B}}) \cap \Re_{\mathcal{B}} = \varnothing$, the* semantics *of CLS is the least transition relation $\rightarrow$ on terms closed under $\equiv$, and satisfying the* *following inference rules:*

$$\frac{(P_1, P_2) \in \mathcal{R} \quad P_1\sigma \not\equiv \epsilon \quad \sigma \in \Sigma}{P_1\sigma \rightarrow P_2\sigma}$$

$$\frac{T_1 \rightarrow T_2}{T \,|\, T_1 \rightarrow T \,|\, T_2} \qquad \frac{T_1 \rightarrow T_2}{(B)^L \rfloor T_1 \rightarrow (B)^L \rfloor T_2}$$

$$\frac{(BP_1, BP_2) \in \mathcal{R}_{\mathcal{B}} \quad BP_1\sigma \not\equiv \epsilon \quad \sigma \in \Sigma}{BP_1\sigma \rightarrow_{\mathcal{B}} BP_2\sigma}$$

$$\frac{B_1 \rightarrow_{\mathcal{B}} B_2}{B \,|\, B_1 \rightarrow_{\mathcal{B}} B \,|\, B_2} \qquad \frac{B_1 \rightarrow_{\mathcal{B}} B_2}{(B_1)^L \rfloor T \rightarrow (B_2)^L \rfloor T}$$

where $\rightarrow_{\mathcal{B}}$ is a transition relation on branes, and where the symmetric rules for the parallel composition of terms and of branes are omitted.

As we previously mentioned, CLS+ will be used in the formalization of explicit maps. Our choice is due to the possibility offered by CLS+ of modeling each population of molecules as a parallel composition of sequences, whether it appears inside a membrane or onto the membrane itself. Moreover, as shown in [29], CLS+ can be translated in CLS: the idea is to represent a membrane with two membranes, one enclosed into the other, the space in between representing the fluid space of the membrane

2.4 The Calculus of Linked Looping Sequences

We remark that in the present work, LCLS will be used in studying a possible formalization for combinatorial maps, where we are interested in interactions at domain level.

In a previous section, we have seen that, in general, CLS is not suitable for expressing domain bindings. That gave the motivation to the authors for extending the calculus with a syntax and semantics for describing links between two different alphabet elements, that is LCLS. For example, in LCLS, the binding wanted in the previous section 2.2.2 would be denoted by $a^1 \cdot b \,|\, c^1 \cdot d$.

Thus, LCLS introduces labels on basic symbols as a new construct. As seen in the example, a label is written as an index at the right top of a basic symbol. For simplicity‘s sake, labels consist of natural numbers. The followings are the basic principles of LCLS:

- two symbols in a term, having the same label, represent a domain binding;

- a basic element can have no more than one label, which means that a single domain can interact at most with one other domain;
- in a term, modeling the whole system, there can not be labels appearing only once, that is, links must be complete: the presence of unmatched labels within a term means that the term describes only a portion of the modeled system, and that the term is indeed a subterm of a term representing the whole system, which will contain a matching label for that symbol;
- links must respect compartments created by membranes: two species can be linked together, if and only if they belong to the same compartment, which means that these species "can see each other" and are not separated by any membrane; therefore, elements inside a membrane can be linked either to elements inside the membrane or to elements on the membrane itself and elements inside a looping sequence can not be linked to elements outside.

The following is the syntax of terms of LCLS.

Definition 2.9 (Terms)**.** Terms T *and* Sequences S *of LCLS are given by the following grammar:*

$$\begin{array}{rcl} T & ::= & S \;\big|\; (S)^L \rfloor T \;\big|\; T \,|\, T \\ S & ::= & \epsilon \;\big|\; a \;\big|\; a^n \;\big|\; S \cdot S \end{array}$$

where a is a generic element of $\mathcal{E}$, and n is a natural number. The infinite set of terms is denoted by $\mathcal{T}$, and the infinite set of sequences is denoted by $\mathcal{S}$.

In order to consider as equivalent two syntactically different terms, the same structural congruence relation defined for CLS is adopted.

In addition, two terms which differ only for the name of their links are considered as equivalent, since they represent indeed the same term: the name given to links of a well–formed term, is a marginal detail, not affecting the semantics of the modeled system. For instance $a^1 \cdot b^1 \cdot c^2$ is considered equivalent to $a^2 \cdot b^2 \cdot c^1$. To this purpose, a renaming function and an equivalence relation are defined, the so called, α–renaming function and α–equivalent relation, respectively. An α–renaming function merely maps each value for labels to a new one. The α–equivalence relation has been introduced due to the possibility given by LCLS of reusing the same label in different compartments: this requires a slightly more complicated definition (see appendix A).

In order to define patterns of LCLS, a construct for labeled element variables is introduced in sequence patterns: variable x^n denotes a possible labeled elements.

A formal method is given to check whether a term is *well–formed*, i.e. the term respects the basic principles listed previously, on which LCLS relies. To this purpose, the notion of *compartment* and of *top–level compartment* of a pattern are given:

Compartment: given a pattern, each of its subpatterns, which is contained in a looping sequence and whose content is ignored, is a compartment;

Top–level compartment: given a pattern, the portion of it, which is not contained in any looping sequence, is the top–level compartment of the pattern.

Informally, "an LCLS pattern is well–formed if and only if a label occurs no more than twice, and two occurrences of a label are always in the same compartment" [30]. A *type system* is used to derive the well–formedness of a pattern. The type system is defined by a set of inference rules, whose conclusion has the form $(N, N') \models P$, where N and N' are two sets of natural numbers (its definition can be find in appendix A).

N is the set of labels appearing twice in the top–level compartment of P;

N' is the set of labels appearing once in the top–level compartment of P.

We write $\models P$ if there exist $N, N' \subset \mathbb{N}$ such that $(N, N') \models P$, and $\not\models P$ otherwise.

Figure A.2 shows some example of well–formed and non–well–formed patterns in LCLS. To our purpose, we are particulary interested to the third example, which shows that a link can connect al most two compounds.

Once introduced the type–system, the well–formedness of terms can bedefined in the following way:

Definition 2.10 (Well–Formedness of Terms)**.** *A term T is* well–formed *if and only if $\models T$ holds.*

Due to the fact that a term variable in LCLS could be instantiated with terms containing labeled elements, an instantiation function σ for LCLS patterns can not merely substitute each occurrence of a variable X with $\sigma(X)$: its definition is slightly more complicated and must rename labels, if necessary (see appendix A)

The definition of rewrite rule is just the same as in CLS.

A main problem in giving a formal semantics for LCLS is due to the possibility of instantiating variables with terms containing only half a link: two patterns describing a rewrite rule must preserve this single labels (see appendix A for further details). The following is an operational semantics of LCLS.

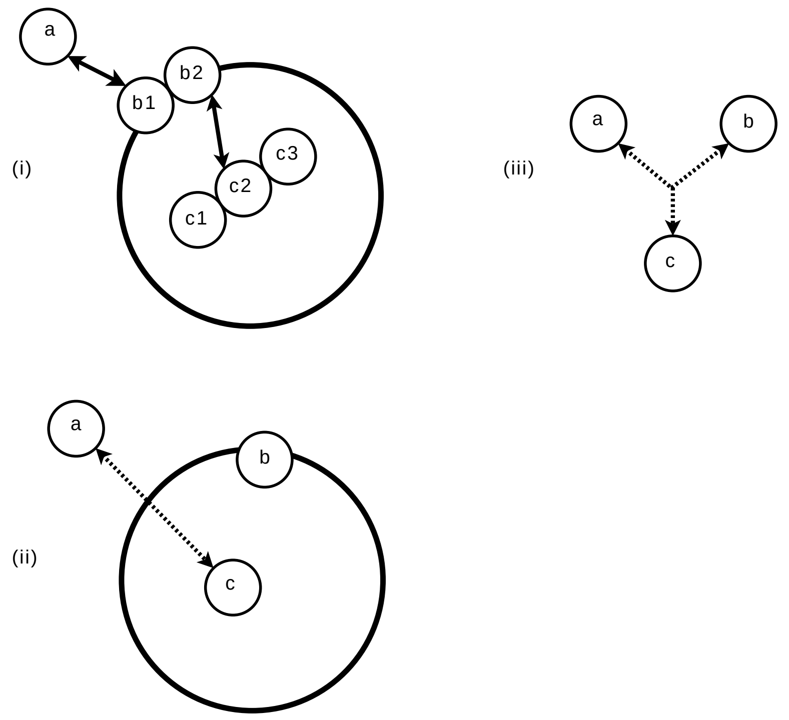

Figure 2.3 Example of well–formed and non–well–formed patterns in LCLS: (i) represents $a^1 \mid \big(b1^1 \cdot b2^2\big)^L \rfloor c1 \cdot c2^2 \cdot c3$; (ii) represents $a^1 \mid \big(b\big)^L \rfloor c^1$; (iii) represents $a^1 \mid b^1 \mid c^1$.

Definition 2.11 (Semantics). *Given a set of rewrite rules $\mathcal{R} \subseteq \Re$, such that $\mathcal{R} = \mathcal{R}^{CS} \cup \mathcal{R}^{CU}$ with $\mathcal{R}^{CS} \subset \Re^{CS}$ and $\mathcal{R}^{CU} \subset \Re^{CU}$, the* semantics *of LCLS is the least transition relation $\rightarrow$ on terms closed under $=_\alpha$, and satisfying the following inference rules:*

$$\textit{(app)} \quad \frac{P_1 \mapsto P_2 \in \mathcal{R} \quad \sigma \in \Sigma \quad \alpha \in \mathcal{A} \qquad (N_1, N') \models P_1\sigma \quad (N_2, N') \models P_2\sigma \quad P_1\sigma \not\equiv \epsilon}{P_1\sigma\alpha \rightarrow P_2\sigma\alpha}$$

$$\textit{(par)} \quad \frac{T_1 \rightarrow T_1' \qquad \models T_1 \,|\, T_2 \qquad \models T_1' \,|\, T_2}{T_1 \,|\, T_2 \rightarrow T_1' \,|\, T_2}$$

$$\textit{(cont)} \quad \frac{T_1 \rightarrow T_1' \qquad \models (S)^L \rfloor T \qquad \models (S)^L \rfloor T'}{(S)^L \rfloor T \rightarrow (S)^L \rfloor T'}$$

where the symmetric rule for the parallel composition is omitted.

Rule *(app)* describes the application of a rule: as soon as the two terms, obtained by instantiating the rule patterns, are well–formed and preserve labels occurring only once, we can make a transition between them;

Rule *(par)* propagates the effect of a rewrite rule application to the parallel context, the parallel composition must be well–formed;

Rule *(cont)* is the same as *(par)*, but in case of looping context.

2.5 Stochastic CLS

Biochemical reactions are inherently stochastic biological phenomena. In order to be able to describe the quantitative aspects of chemical reactions, like time and speed, Barbuti et al. developed a stochastic extension of the Calculus of Looping Sequences. A comprehensive formalization of the calculus is given in [28]. The stochastic extension is based on CLS, thus, since the present work focuses on CLS+ and in minor part on Linked CLS, we will not recall Stochastic CLS formally. However, we think that a formal stochastic extension of CLS+ should be straightforward and a stochastic extension for LCLS should be feasible too. We will try to explain how the concept of reaction rate is added to the calculus and the mechanism for choosing the next action to perform and its time.

A model in CLS is given by a CLS term and a set of rewrite rules. The stochastic extension of the calculus assumes that a *rate* is combined with each rewrite rule, modeling the speed of the activity described by the rule. This rate can be considered as equivalent to the rate constant which could be assigned to each reaction in a molecular interaction map, and in general with a chemical reaction. A stochastic ground rewrite rule is hence defined as a triple (T_1, T_3, p), denoted by $T_1 \overset{p}{\mapsto} T_2$, where p represents the *rewriting rate constant.*

Given a term, a ground rewrite rule could be applicable to different subterms (reactants) of the term, hence, according to which subterm the rule is applied, the application of the rewrite rule to the term can result in different terms. At a given time, the choice of which rule to apply, among the many applicable rules, depends on the rate of each rule and the number of subterms to which the rule can be applied, which can be considered equivalent to the number of possible combination of the existing reactants of the considered reaction in the modeled biological system. In this way, Stochastic CLS follows the mass–action law, since the speed and probability of a reaction to happen (that is, rewrite rule to be applied), depends on the concentrations of its reactants (that is, number of subterms to which the rule can be applied)

Computing the number of subterms to which a rule can be applied is not an easy task. First, the set of all the ground rules that can be applied to a given term must be computed. Then the *application cardinality* of each ground rewrite rule must be calculated. The application cardinality is roughly the number of times a ground rewrite rule can be applied to a term. This procedure counting the application cardinality must take into account that the ground rewrite rule could be applicable to different subterms (reactants) of the term and its application could hence result into different terms. Thus two applications of the same ground rewrite rule to a term, which yields to different terms, must be considered as two different reactions.

In the end, the stochastic semantics assigns to each possible transition a rate (*transition rate*), which characterizes the stochastic behavior of the activity modeled by the rewrite rule. This rate is obtained as the product of the rewriting rate constant and the application cardinality of the rule used to perform the transition. The higher is the transition rate, the higher is the probability for that transition to happen.

As shown in [28], the model obtained by applying the stochastic semantics to a given term, is essentially a *Continuous–Time Markov Chain.*

As we have seen, given the CTMC, a classical simulation procedure, corresponding to Gillespie's simulation algorithm can be applied. A configuration (state) of the simulation model is a triple $(T, \mathcal{R}, t)$, where T is the actual term, $\mathcal{R}$ is a finite set of stochastic rewrite rules and $t \in \Re^{\geq 0}$ is the time of a global clock. Given a configuration, starting from the given term, there is a finite set of transitions, each labeled with a transition rate r_i and the rewrite rule which is applied. The addition of all the transition rates

$E = \sum\limits_i r_i$ gives the *exit rate* of the given configuration. The time at which the next transition will occur is chosen with an exponential distribution with exit rate E as parameter. The transition to be performed is then chosen with probability $\frac{r_i}{E}$.

Chapter 3

A Formal Semantics for MIMs

In previous chapters, we have seen how important is nowadays for biologists to have a standard notation to represent interactions at molecular level within a cell environment, such as formation of complex molecules through bindings, and membrane hierarchies. This standard notation should serve as a starting point for a possible computer simulation over time, therefore it should also keep track of kinetic constants, whose value affects the evolution of the modeled system in time. To this purpose, we have shown a diagrammatic notation for molecular interactions, namely the Molecular Interaction Maps (MIMs), which are able to describe formation of complexes and allows sketching of membranes, without omitting quantitative aspects. We explained how formation of complexes in a MIM can be interpreted in different ways: explicitly and combinatorially. Then, we have described a formal language to model biological systems, namely the Calculus of Looping Sequences (CLS), whose semantics is basically a transition system: a state is represented by a term, describing the qualitative aspects of a biological model, such as presence of certain molecules or complexes and a possible membrane structure; possible transitions are defined by a set of rewrite rules, describing the possible evolution of the modeled system. We dwelled upon three extensions of CLS: CLS+, LCLS and Stochastic CLS, this last adding time and rates for modeling quantitative aspects of the biological system.

We already gave some anticipation on how we will give a possible semantics for Molecular Interaction Maps. Now, we can proceed to reveal the details of our formal semantics. The semantics is given by translation of MIMs into some Calculus of Looping Sequences: a map is modeled with a term, representing the initial state of the map, and a set of rewrite rules, describing all the possible transitions between different configurations. Given a map, we try to define a formal semantics which unambiguously establishes the initial term and the set of rewrite rules which model the map.

First, we will define a formal method for representing, with a term in the chosen calculus, the initial configuration and topology of a map, that is all the molecules and complexes which could exist at the beginning of a possible simulation of the modeled system and, when necessary, the membrane structure. It is reasonable to assume that, initially, the system modeled by a MIM contains only elementary species. Complex species will eventually appear as consequence of the execution of some reaction.

Then, we define a formal method for constructing a set of rewrite rules, representing interaction symbols, both reactions and contingencies, and which defines the possible evolution of the initial term over the time. MIM's authors pointed out in [10] that contingencies might introduce some ambiguity in the semantics of a map, thus, they suggested to avoid them in case the map is needed for computer simulation. Giving a formal semantics to contingency symbols allow them to be used in computer simulation. Our formalization of contingencies is possible by giving them a precise meaning. We will give further details on how contingencies are interpreted in the next section.

The whole formalization procedure requires an intermediate encoding of MIMs, through structures made of set of tuples, each set describing some MIM component. We try to keep this intermediate encoding close to the tables already used by biologists to describe MIMs, like the reaction table and the molecular species table: in this way we hope that biologists could become more familiar with our encoding of MIMs into some CLS, and maybe using our encoding to perform some simulation using the already existing stochastic CLS simulator.

In giving a formal semantics, we keep the distinction among explicit and combinatorial maps.

We first give a detailed semantics for *explicit* maps. In this case, the formalization is given through translation into CLS+. The following are the intermediate structures used to encode the initial CLS+ term.

Multiset of Species, each tuple in this set corresponds to a certain initial species in the MIM and describes the name of this species, and its quantity;

Tree Structure, each tuple in this set corresponds to a certain membrane in the MIM and describes the identification number of this membrane, the set of membranes which are directly contained into the one at issue, the Multiset of Species appearing onto this membrane and the Multiset of Species appearing into this membrane; this tree structure is a rooted tree, reflecting the membrane nesting;

Given a map, we proceed formally defining a Tree Structure which represents the membrane hierarchy and the set of species which initially appear in the map. Then, given this Tree Structure, we define a translation function that constructs the corresponding CLS+ term, in which each sequence represents a species and the nesting of looping reflects the membrane structure.

Once we have an initial term, we concentrate on the definition of a set of rewrite rules, reflecting the semantics of interaction symbols appearing in a certain map. The following are the intermediate structures used to construct the CLS+ set of rewrite rules.

Membrane Structure, each tuple in this set corresponds to a certain membrane in the MIM and describes the set of membranes which are directly contained into the one at issue; hence, the membrane structure is a rooted tree, reflecting the membrane nesting, but, differently from the tree structure, it does not keep information about species;

Set of Species, each tuple in this set corresponds to a certain species in the MIM and describes the name of this species, its position, with regard to the membrane hierarchy, and its quantity; this structure practically adds information about position to the Multiset of Species;

Set of Contingencies, each tuple in this set corresponds to a certain contingency line in the MIM and describes the species required for the contingency to be satisfied (a Set of Species) and the reaction rate at which the pointed reaction can happen once the contingency at issue is satisfied; note that a Set of Contingencies will be combined with a certain reaction.

Set of Reactions, each tuple in this set corresponds to a certain reaction line in the MIM and describes the species involved in the reaction (reactants and products, which are indeed Set of Species), the Set of Contingencies affecting the reaction at issue and a Membrane Structure, which represents the minimum sub–tree of the original membrane hierarchy containing all the species involved in this reaction.

We proceed constructing the Membrane Structure representing the membrane hierarchy of the given map. Then, we formally define the Set of Reactions for the given map, combining a formal description of the proper Set of Contingencies with each reaction. Finally, we define a translation function that, given a Set of Reactions, constructs a set of CLS+ rewrite rules, each describing a reaction.

Each rewrite rule will be a triple containing a CLS+ pattern describing the reactants, a CLS+ pattern describing the products and a rate function: this function describes which rate constant should be assigned to the rewrite rule, depending on which contingency is verified in a given state. In order for a certain contingency to be verified, some variable in the patterns of the rewrite rule must match with the sequences representing the required contingency species. Further details on how contingencies are interpreted are given in the next section.

We remark that an explicit map is generally equipped with a reaction table (describing reactants and products of each labeled reaction) and a molecular species table (describing labeled species in the map). As we have remarked in section 1.2.4, there can be some dummy reaction symbols that actually appear in a map, but that in practice are not entitled to happen. Such dummy reactions do not appear in the reaction table and moreover, to highlight the fact that in practice they are not supposed to be part of the model, they are usually not labeled in the map. Thus, in producing the Set of Reactions, we will refer to labeled reactions and hence to the reaction table: in practice, we will produce a reaction tuple for each entry in this reaction table. Therefore, even though a reaction is depicted in the map, if the designer demonstrates his will for such a reaction not to be considered in the model, by not labeling it, we respect his/her willing by not including this reaction in the CLS model. In this way, if one is interested in modeling only part of the given map, he/she does not have to build a new map containing the wanted reactions and species: he/she can simply specify a sub–map by labeling the only elements of interest. Thus, we give the possibility to model only a part of the map. Allowing this sub–map to be simulated independently from the rest of the map. The same considerations can be done for species: we assign a CLS name only to labeled species.

We would like to note, that we could also let the our formal translation into a CLS model to "mechanically" encode each and every reaction symbol appearing in the map. In this case we would need a way to specify that some reactions are inactive: we could permanently inhibit them or we could give them a null rate constant. In this work, we chose to simply not insert them in the CLS model.

At the end of this section, we describe how part of the explicit map for the well note EGFR Signalling Pathway can be translated into a Stochastic CLS+ model, following our formal semantics. We point out how, giving a formal semantics to contingencies symbols, allows them not to be totally excluded, in depicting explicit maps aimed at computer simulation. In particular, we show how an enzymatic catalysis appearing in the EGFR map can be represented by defining one reaction with a contingency, without having to produce three explicit reactions for it.

In the second part of this chapter, we discuss, in a less formal way, how a possible semantics can be given for *combinatorial* maps. In this case, the study of a possible semantics is done by translation into Stochastic LCLS+.

We will formally define how an initial term could be constructed: the procedure is similar to what we do in the explicit interpretation, with some minor difference, due to the fact that in a combinatorial map we are interested in interactions at domain–site level. The ability of modeling interactions at domain level will be useful in describing formation of complexes in a combinatorial interpretation.

As we anticipated in the previous chapter, the basic idea behind our possible formalization of "combinatorial" complexes is that, if we can name each domain or interaction site univocally in the map, and assuming that a reaction line in a MIM connects two disjoint domains, then we can identify a reaction with the pair of the corresponding interacting domains. We will not give a formal semantics for reactions, instead, we will show what difficulties are encountered in describing reactions with Stochastic LCLS+ rewrite rules, due to the presence of compartments created by membranes, and guidelines for a possible solution: in particular, we will stress the difficulties in modeling a translocation of some complex molecule from a compartment to another. We give some idea about a possible solution. We observe that, anyway, a formal semantics through translation into Stochastic LCLS+ can be easily given for "flat" combinatorial maps, which do not have a membrane structure and we provide some guidance for the construction of a possible set of rewrite rules in this case. Finally, we show an alternative interpretation of contingencies at domain level.

In our formalization, we will try to keep name of species in the CLS model closest as possible to those appearing on the map. In naming species on the map, we suggest, when possible, to follow the biochemical nomenclature, by which a complex formed by the binding of A and B is denoted with $A{:}B$, a phosphorylated species A is denoted by pA and so on[1]. This would make the term representing the biological system more friendly from a biological point of view. Of course, as soon as this nomenclature could become too clumsy, due also to the length of a string representing a possible complex, it is recommended to abbreviate the name, not to go against our good intentions.

In both formalizations, we will assign to each membrane appearing in a map an univocal identification number in $\mathbb{N}^{\geq 0}$. We will assume the whole system to be surrounded by an auxiliary membrane 0, in case this outer membrane does not exist in the map, it will be added. The reason for this assumption is that in our formalization we want to deal with a single rooted tree representing the whole membrane hierarchy: if we do not assume the presence of an outer membrane containing the whole hierarchy, the membrane hierarchy would be indeed represented by a forest of trees, which would be not always as easy to manage as a rooted tree. In Figure

[1]Note that a phosphate **P** is not modeled as a species, since in general, there are thousands of such particles dispersed all over a cell (its one of the so–called *ubiquitous* molecules, which comprehends methyl, acetyl, ubiquitin, etc.).

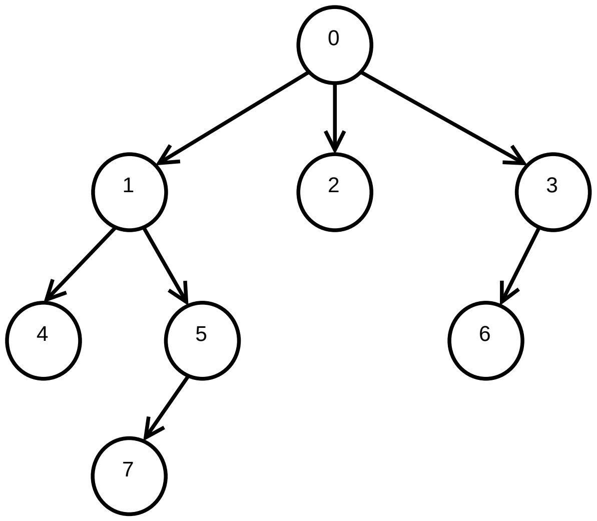

Figure 3.1 Tree representing a possible membrane hierarchy: nodes are indexed in breadth–first fashion order.

3.1 you can note how the membrane hierarchy gives rise to a tree rooted in 0, where a node with some children stands for a membrane with some inner membrane. We recommend to name membranes in a breadth–first fashion order, as it is done in Figure 3.1[2]

Before starting with the explicit and combinatorial formalizations, we give some guidance on how we interpret interactions symbols, focusing on contingency symbols.

3.1 Interpretation of Interaction Symbols

For each reaction in the map, we have a set of zero or more reactants and a set of zero or more products. To each reaction should be matched the set of the contingencies affecting its rate constant. A contingency is an arrow connecting a species to a reaction, describing the influence of the species on the rate of the pointed reaction.

Informally, a contingency can be seen as the requirement of some species to exist in a particular place for the reaction to happen with a certain rate constant. Informally, a reaction can happen in each state where the

[2]Breadth–first fashion order allows us to state that each sub-tree of such a membrane structure is rooted at its minimum node, however, as we will see, the order is not mandatory, since we can always identify the root as the only node with no children. Another possible nomenclature for membranes, which carries information about parent nodes, is one that assigns to a membrane i_n a name of the kind $0.i_1.\ldots.i_{n-1}.i_n$, reflecting the fact that membrane i_n is contained into i_{n-1}, which is contained in i_{n-2} and so on, until the root 0. This nomenclature could result in long names for membranes.

reactants are presents in the required places. If this condition is verified, we still have to compute the rate at which the reaction could possibly take place. The rate of the reaction depends on the concentrations of the reactants and on which of its contingencies are verified. For instance, a greater concentration should increase the eventual rate of the reaction; in case of existence of some inhibiting species, the reaction should happen at a reduced rate; while, in case of existence of some stimulating species, the reaction should possibly happen at a higher rate. On the other side, a reaction should be possible, independently on whether other species, which do not take part in the reaction (third species), exist or not.

The interpretation of contingencies in this work is quite similar to what has been presented in [29]In absence of contingencies a reaction could take place with a certain rate constant[3] .

The main idea behind contingencies is the following: a contingency simply sets the rate constant of the pointed reaction with a new one. If a certain contingency is verified, the reaction will get the new corresponding rate constant. Thus, a **stimulation** will set the rate constant at which the reaction can happen with a higher one. **Catalysis** is quite similar, the main difference is that in case of reversible binding a stimulation will increase the rate for complexation and decrease the one for decomplexation, while a **catalysis** will increase both rates. In case a certain species is a **requirement** for a reaction to happen, the reaction rate of the reaction will be zero, or about zero, until the required species will appear in the required place, then the reaction rate will be set to a value greater then zero. In case a certain species inhibits a reaction, the reaction rate of this reaction will become zero or somehow reduced, once the inhibiting species appears in the required place.

Of course, it is possible to have more than one contingency symbol pointing to the same reaction. As we said, each contingency sets the reaction rate with a new one, that must be used in case the contingency at issue is satisfied. What if more than one contingency is satisfied at the same time? In this case, we need to define a new rate constant for each possible combination of contingencies which can be verified at the same time. In order to disambiguate the choice of the rate constant in case of multiple contingency, some state–combination symbols can be used in the map. We show a little example in Figure 3.2. We explain how the two figures should be interpreted.

[3]In this work, we use without distinction “rate constant” and “kinetic constant”. Remark that, in practice, in performing a simulation applying Gillespie’s stochastic Algorithm, a new “simulation constant” is obtained from the kinetic constant and this simulation constant is the one used in practice, when performing the two steps of the algorithm.

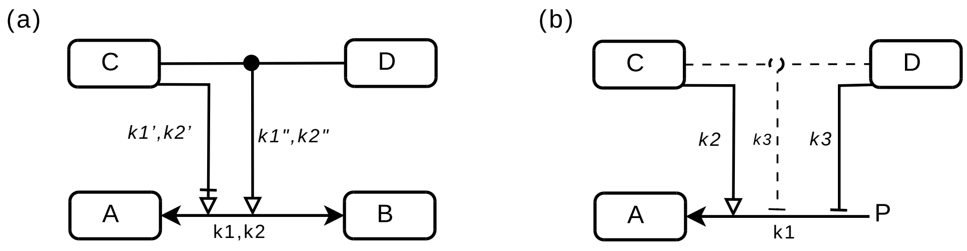

Figure 3.2 Examples of multiple contingencies: (a) **C** is required for the non–covalent binding to happen, **D** stimulates the reaction; (b) **C** stimulates the covalent modification, while **D** inhibits it.

(a) in the absence of **C**, the non–covalent binding between **A** and **B** can not happen, in this case we could assign to $k_1 = 0$ and k_2 a quite high value, denoting that, without **C**, possible complexes **A:B** can easily break down; when **C** is present the complexation can finally happen with a certain rate $k_1' > 0$, if also **D** is present, then the rate for complexation is a higher one ($k_1" > k_1'$); while rate for decomplexation is decreased by the presence of **C** and further decreased by the presence of **C** and **D** together;

(b) In the absence of **C** and **D**, the covalent modification of **A** can happen at a rate of k_1; when **D** exists, whether **C** exists or not, the covalent modification can not happen, thus the new rate constant k_3 will be equal to zero. Therefore, specifying the case in which both **C** and **D** exist is not necessary (for this reason we represented it with a dashed line); **C** can stimulate the modification only in the absence of **D**, in this case we will have a rate constant $k_2 > k_1$.

In practice, each of the four contingency symbol is modeled in the same way, the only difference is in the choice of the new rate constant/s, which is left at human discretion.

In our formalization, a reaction symbol will be represented by one or more stochastic rewrite rules (basically, we need two rules for reversible binding, that indeed can be seen as two reactions).

Recall that a ground rewrite rule in stochastic CLS is a triple of the kind (T_1, T_2, k), describing a possible transition from term T_1 to term T_2 at a rate constant of k. A reaction, in the absence of contingencies affecting it, could be simply modeled through a pair of patterns and a *rate constant*: each pattern will explicitly show species taking part in the reaction, while third species will be eventually matched with general term variables appearing at every position in the patterns, where these third species could possibly appear.

The intermediate representation for *Contingencies* affecting a certain reaction is a set of pairs composed by a set of species and a rate constant.

A reaction affected by contingencies will be modeled by a rewrite rule of two patterns and a *rate function*, (P_1, P_2, f). Term variables will be used in the patterns, to model the environment where the reaction occurs. Having contingencies, they will also model the presence or absence of contingency species. Contingency species affecting the same reaction could be placed onto different membranes in a map. Therefore, we must recollect which term variable models the presence of which contingency.

The rate function will return a different rate constant, depending on which contingency species are present in the instantiations of this variables.

Note that contingencies could be modeled in a more or less accurate way, depending on the rate function definition: we could build a rate function such that the merely presence of a certain contingency species is enough for a contingency to be verified; we could be slightly more accurate and specify that a certain amount of contingency species is needed; being keen, we could assign a different rate constant to increasing ranges of amount of contingency species; being very keen, we could make the rate function directly dependant on the concentration of each contingency species. In our work, we show how the rate function should be defined in general, leaving the level of accuracy up to the designer.

Note that, each reaction, whether it is affected by contingencies or not, has a rate constant that must be used in case none of its contingencies is verified: this rate constant will be also represented by a pair in the set of contingencies, namely the pair formed by the empty set and this rate constant. Thus, in practice, all reactions, in their intermediate encoding, have an associated set of contingencies and formally they are all modeled by a rewrite rule of two patterns and a rate function, but in case of absence of contingencies, the rate function will be indeed a constant.

We show what rewrite rules (indeed, possible transitions) would result for maps of Figure 3.2. We assume the existence of the outer membrane 0. From the map on the left we would obtain:

$$(0)^L \rfloor (X \mid A \mid B) \overset{f_1}{\mapsto} (\epsilon)^L \rfloor (X \mid A{:}B)$$

$$(0)^L \rfloor (X \mid A{:}B) \overset{f_2}{\mapsto} (\epsilon)^L \rfloor (X \mid A \mid B)$$

$$f_1 = \begin{cases} k1' & \text{if } \sigma(X) \equiv C|T \\ & \text{and } T \not\equiv D|T' \\ k1'' & \text{if } \sigma(X) \equiv C|D|T \\ k1 & \text{otherwise} \end{cases} \quad f_2 = \begin{cases} k2' & \text{if } \sigma(X) \equiv C|T \\ & \text{and } T \not\equiv D|T' \\ k2'' & \text{if } \sigma(X) \equiv C|D|T \\ k2 & \text{otherwise} \end{cases}$$

and from the map on the right:

$$\big(0\big)^{L} \rfloor (X \mid A) \overset{f}{\mapsto} \big(0\big)^{L} \rfloor (X \mid A\_P)$$

$$f = \begin{cases} k2 & \text{if } \sigma(X) \equiv C|T \\ & \text{and } T \not\equiv D|T' \\ k3 & \text{if } \sigma(X) \equiv D|T \\ k1 & \text{otherwise} \end{cases}$$

for some terms T, T'. For instance, having $\big(\epsilon\big)^{L} \rfloor (A|A|A|B|B|B|C|D)$ as initial term for the first set of rules, we could have the following transition:

$$\big(0\big)^{L} \rfloor (A|A|A|B|B|B|C|D) \overset{k2"}{\mapsto} \big(0\big)^{L} \rfloor (A{:}B|A|A|B|B|C|D)$$

having $\big(0\big)^{L} \rfloor (A|A|C|D)$ as initial term for the second diagram, we could have this transition:

$$\big(0\big)^{L} \rfloor (A|A|C|D) \overset{k3}{\mapsto} \big(0\big)^{L} \rfloor (pA|A|C|D)$$

Supposing that $k3$ is equal to zero, this means that this transition has probability 0 of happening.

In this examples, the rate function is a naive approximation of the effect of contingencies on reactions: having one or one million instances of a stimulating species yields to the same rate constant. We could instead use an auxiliary function that computes the concentration of each contingency species in variable X and we could make the rate function depending on this concentration. For instance, being $c_1, c_2, \ldots, c_n$ the concentrations of the n contingency species and $k_1, k_2, \ldots, k_n$ the respective kinetic constants, we could define n functions $(f_1(c_1), f_2(c_2), \ldots, f_n(c_n)$ and then the rate function n as a linear combination of these functions $f = k_1(f_1(c_1)) + k_2(f_2(c_2)) + \ldots + k_n(f_n(c_n))$ or some other non–linear combination.

In this work, since our main purpose is giving a formal semantics, for sake of simplicity, we give specifications of rate functions using kind of naive approximation: we will allow the specification of a minimum amount of contingency species to exist for the contingency to be verified.

3.1.1 Contingencies pointing to other contingencies

We know that, in MIM notation, is possible for a contingency to point to another contingency, creating a chain of contingencies, where a final contingency points to a reaction. In our interpretation of this contingency sequence, each subsequence of this sequential combination of contingencies, could be

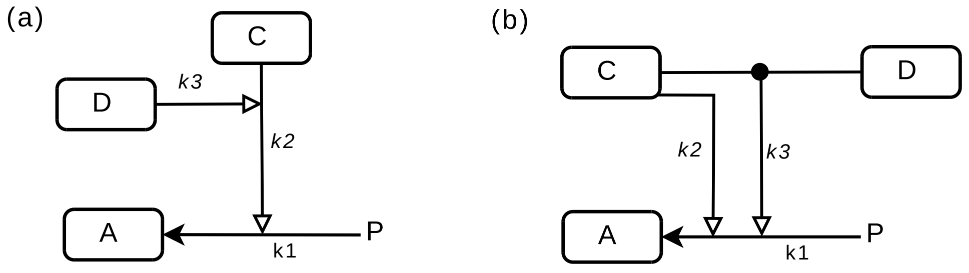

Figure 3.3 Examples of how representing a sequence of contingencies with a state–combination symbol: (a) shows the sequential notation; (b) shows the same contingencies with a state–combination symbol.

collapsed into a contingency which points directly to the final reaction; the species at the tail of this new contingency, will be the state–combination of all the species appearing along the contingency sequence. Figure 3.3 shows a small example clarifying this situation:

(a) **C** stimulates the reaction; **D** stimulates the previous stimulation: in our interpretation, this means that if **D** is present, beside **C**, the reaction is further stimulated, that is $k_1 < k_2 < k_3$.

(b) the same contingencies of (a) represented with a state–combination symbol: **C** stimulates the reaction; the presence of both **C** and **D** further stimulates the reaction.

3.2 Explicit interpretation of a MIM

We will now concentrate on the formalization in CLS+ of a MIM in its explicit interpretation.

Recall that, in this interpretation, an interaction line refers just and only to the two species pointed by the line: it is not possible to apply the interaction to species other to those pointed by the line. Thus, as we observed in section 1.2.4, listing all possible complexes in advance is straightforward.

Recall that we assume that each interaction can happen only in the particular place where it is depicted and that the membrane structure does not change over time.

As usual a model in a Calculus of Looping Sequences for a Molecular Interaction Map, will be composed by an *initial term* and a *set of rewrite rules.* We first describe a formal method to construct the initial term and then we will concentrate on the construction of a possible set of rewrite rules.

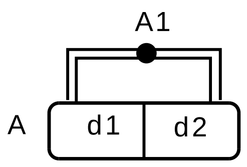

Figure 3.4 Example of named MIM diagram in its explicit interpretation, containing an intramolecular binding. Note how a new name is assigned to the complex representing the molecule whose domains are bound together

3.2.1 Initial Term

Given a map in its explicit interpretation, we need an intermediate encoding for its membrane structure and the possible initial set of species. Then we define a translation function that, given this intermediate representations of membrane structure and set of species, produces the initial CLS+ term representing it.

In its explicit interpretation, it is possible to list in advance all the possible molecular species (both elementary and complex) that could eventually arise in the model representing the map. Thus, given a map, we proceed assigning an univocal CLS+ name to each possible molecular species. We assume that these name can be derived using the CLS+ syntax of sequences, that is:

$$S \ ::= \ \epsilon \ \big| \ a \ \big| \ S \cdot S$$

where a is a generic element of $\mathcal{E}$, which is a possibly infinite alphabet of symbols ranged over by $a, b, c, \ldots$. Thus, names are readily made CLS+ elements and, in the following, we assume that names appearing in a "named" map are CLS+ elements.

As we noted in section 2.2.1, in general, a molecular species could be represented by a CLS+ sequence. A multi–domain molecular species could be represented, as the sequence composed by its name followed by the name of its domains, for instance species **A** of Figure 3.4, having two domains **d1** and **d2** could be represented in CLS+ as $A \cdot d_1 \cdot d_2$. It is worth noticing that, in this interpretation, multi–domain species could be merely described as a single alphabet element: a species with an intra–molecular bond can be seen as a new kind of species and can be described by a new univocal name. As you can see, in Figure 3.4, we adopted this naming strategy and we just assigned a different name to species **A**, when it is in a state where the intra–domain bond exists, that is $A1$.

Thus, we could limit the representation of species to CLS+ alphabet elements, however, if preserving information about intra–domains could make the model more readable from a biologica point–of–view, species could

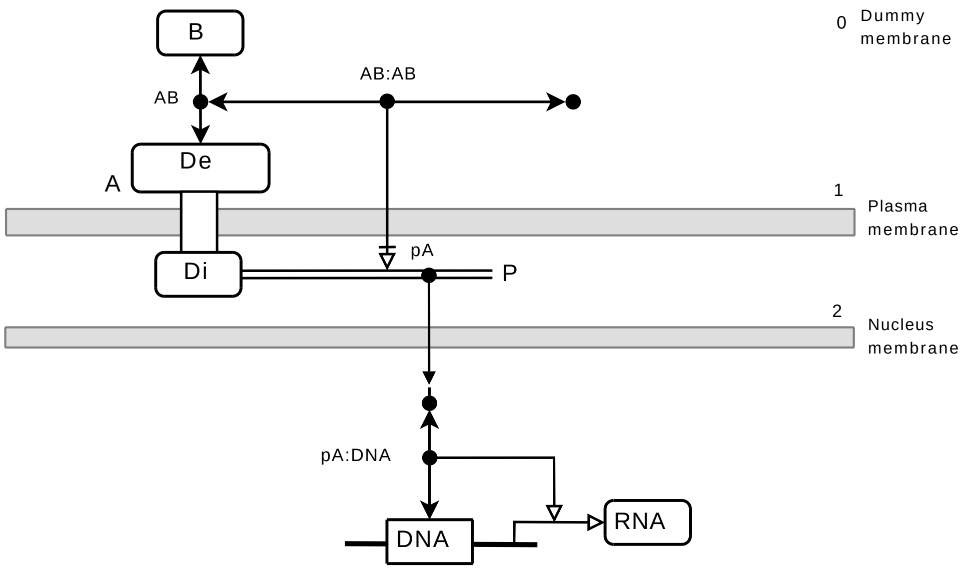

Figure 3.5 Example of named MIM diagram in its explicit interpretation

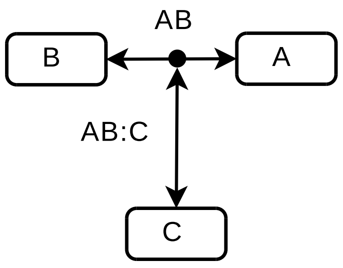

Figure 3.6 Example of named MIM diagram in its explicit interpretation, each complex is assigned an univocal name, even if this complex is not represented by a node in the map.

still be represented by a sequence describing its intra–domains.

The membrane hierarchy of a map is statical, that is, membranes will be the same represented in the map, throughout the whole system evolution. Thus, we assign each membrane an unambiguous identification number in $\mathbb{N}^{\geq 0}$, following the assumption that the whole system is surrounded by a dummy membrane named 0.

In map of Figure 3.5, we show how, each of such components (species and membranes), appearing in the map used in section 1.2.7 as example, can be named. Note that in case of elementary species, the CLS+ name we assigned to them coincides with the one of the original map, while in case of complex species, most of the times, we created a new name.

Note that we assign a name to each possible complex, whether this is represented by a node in the map or not. For instance, in Figure 3.6, complex resulting from the binding of **AB** and **C**, does not further interact and hence is not represented by a node in the map, but the CLS+ name $AB{:}C$ is still assigned to it.

We define an intermediate encoding of the membrane hierarchy and the species initially located in each membrane as a tree rooted in 0. Recall that we assumed the whole MIM to be contained in a dummy membrane of index 0. For each membrane, we keep track of its children membranes and which species are either on top of it or contained by it. We first give the formal definition of Multiset of Species, which will be used to describe for each membrane, which species are contained by it and which are on it. Recall that $\mathcal{S}$ is the set of all possible CLS+ sequences.

Definition 3.1 (Multiset of Species, MSS)**.** *A* Multiset of Species MSS *is a pair of the form:*

$$\langle SS, m\rangle \in 2^{\mathcal{S}} \times [\mathcal{S} \rightharpoonup \mathbb{N}]$$

where:

$SS \in 2^{\mathcal{S}}$ *is the set of CLS+ sequences, each corresponding to some species;*

$m \in [\mathcal{S} \rightharpoonup \mathbb{N}]$ *is the quantity function, which is a partial function describing the quantity of each species in* SS*.*

We denote with $\mathcal{MSS}$ *the infinite set of all possible Multisets of Species.*

In alternative, a multiset could be seen as a set of pairs of the kind (species name,quantity), that is $\langle S, q\rangle \in \mathcal{S} \times \mathbb{N}$. We will use this representation when it could be of more practical use.

The following is the formal definition of the Tree Structure describing initial species and membranes in a MIM.

Definition 3.2 (Tree structure of a MIM, TS)**.** *Given a MIM, its* Tree Structure TS *is a set of tuples of the form:*

$$\langle i, C, MS, IS\rangle \in \mathbb{N} \times 2^{\mathbb{N}} \times \mathcal{MSS} \times \mathcal{MSS}$$

where, each tuple, that is a node in the tree, corresponds to a membrane in the MIM and:

$i \in \mathbb{N}$ *is the membrane identification number; we assume* $i = 0$ *the id of the outmost membrane, which will be added to the MIM, in case it does not exist;*

$C \in 2^{\mathbb{N}}$ *is the set of identification numbers of the membranes which are immediately contained by the membrane at issue. If there are no such membranes, C is equal to the empty set $\varnothing$;*

$OS \in \mathcal{MSS}$ *is the multiset of species appearing onto the membrane at issue (Over membrane Species); if there are no species onto the membrane, OS will be equal to $\varnothing$;*

$IS \in \mathcal{MSS}$ *is the multiset of species appearing into the membrane at issue (Inner membrane Species); if the membrane contains no species, IS will be equal to $\varnothing$;*

We denote with $\mathcal{TS}$ the set of all possible MIM Tree Structures.

For instance, the tree structure representing a possible initial situation, in which only elementary species are presents, of the map of Figure 3.5 will be the following set of tuples:

$$\begin{aligned} TS = \{C_0 &= \langle 0, \{1\}, \varnothing, \{(B, 2)\}\rangle, \\ C_1 &= \langle 1, \{2\}, \{(A, 3)\}, \varnothing\rangle, \\ C_2 &= \langle 2, \varnothing, \varnothing, \{(DNA, 1)\}\rangle\} \end{aligned}$$

As we can see, we can simply omit species whose quantity would be 0. We adopted the alternative representation of multiset, as pairs of (species_name, quantity), for sake of simplicity.

Now, we need to define a translation function that, given a Tree Structure, constructs the corresponding CLS+ term. For each tuple in the tree structure, the translation function constructs a looping sequence, representing the membrane at issue, whose loop consists of the parallel composition of the CLS+ sequences corresponding to species in the set OS and that contains the parallel composition of the recursive translation of its inner membranes and the CLS+ sequences corresponding to species in the set IS. The membrane identification number will be translated as an univocal CLS+ alphabet element, which will be put in parallel with the other species appearing onto the membrane.

We first define an auxiliary translation function, namely ϕ, that, given a multiset of species, constructs a sequence for each species and put them together in a parallel composition. Recall that $\mathcal{T}$ denotes the infinite set of all possible CLS+ terms.

Definition 3.3 (Multiset of species Translation Function, ϕ)**.** *Given a multiset of species (SS, m), the function $\phi : \mathcal{MSS} \rightarrow \mathcal{T}$ constructs the correspondent CLS+ parallel composition $T = \phi(SS, m)$, according to the following* *definition:*

$$\phi(\varnothing) = \epsilon$$

$$\frac{m(\widetilde{x}_i) = n_i \qquad i = 1, 2, \ldots, q}{\phi\langle\{\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_q\}, m\rangle = n_1 \oplus \widetilde{x}_1 \,|\, n_2 \oplus \widetilde{x}_2 \,|\, \ldots \,|\, n_q \oplus \widetilde{x}_q}$$

where $\widetilde{x}_i$ are generic LCLS Sequences, $n \oplus T$ stands for a parallel composition of n times T, that is $T \,|\, T \,|\, \ldots \,|\, T$ of length n, where T is a generic LCLS Term. Note that $0 \oplus T \equiv \epsilon$.

Notice that ϕ applied to an empty set of species returns the empty sequence. Some example of ϕ application could be the following:

$$\phi\langle(A, 2), (B, 3)\rangle = A \,|\, A \,|\, B \,|\, B \,|\, B$$

At this point, we would like to make the reader notice the advantage in using CLS+ over simple CLS: in CLS+, both species appearing onto a membrane and species appearing into a membrane can be represented with a parallel composition of sequences, thus we can use the previous Multiset of Species Translation Function both for translating the set of species onto the membrane and the set of species appearing into a membrane; while, in CLS we would have needed a second translation function, which would have built a single sequence representing the set of species appearing onto the membrane. Moreover, having all the species concatenated in a single sequence, could have led to a loss of information in case some species is indeed represented as a sequence of more than one symbol (we noted in the previous section, how in practice, we could have named species using only alphabet elements): in particular we would have lost the point of division between two species, and probably we would have needed a proper alphabet element working as separator of two different species; moreover, we would have lost full commutativity of species onto a membrane (recall that looping sequences in CLS are only entitled to rotate).

The following is the formal definition of the translation function that, given a Tree Structure, constructs the corresponding CLS+ term.

Definition 3.4 (Tree Structure Translation Function)**.** *Given a tree structure $TS \in \mathsf{TS}$ and being $N = \langle 0, C, \varnothing, IS\rangle \in TS$ the* Tree Structure Translation Function $[\![\cdot]\!]_T : \mathsf{TS} \to \mathcal{T}$ *is defined as following:*

$$[\![TS]\!]_T \mapsto [\![N]\!]$$

where the function $[\![\cdot]\!]$ *is defined by the following rules:*

$$1.\quad [\![\langle i, \varnothing, MS, IS\rangle]\!] \mapsto \big(i \,|\, \phi(MS)\big)^L \rfloor\, (\phi(IS))$$

$$2.\quad \frac{N_j = \langle c_j, C_j, MS_j, IS_j\rangle \in TS \qquad T_j = [\![N_j]\!] \qquad j = 1, 2, \ldots, p}{[\![\langle i, \{c_1, c_2, \ldots, c_p\}, MS, IS\rangle]\!] \mapsto \big(i \,|\, \phi(MS)\big)^L \rfloor\, (\phi(IS) \,|\, T_1 \,|\, T_2 \,|\, \ldots \,|\, T_p)}$$

Note that the translation function utilizes an auxiliary function that operates on a single tuple. We could have also defined the translation function as the only $[\![\cdot]\!]$. However, the auxiliary function makes simpler a possible call to the translation function: in fact, given a Tree Structure, in calling the translation function on it, we do not need to retrieve the root node of the Tree Structure and then call the function on this tuple, while this would be necessary in case we defined the translation function as the only $[\![\cdot]\!]$. Since we assume the Tree Structure being a tree rooted in 0, we can be sure that, indeed, a node 0 representing the root of the tree exists. Thus, we can simply apply the auxiliary function to the root 0. We anticipate to the reader that, in some future similar definitions, (precisely in the semantics of rewrite rules), we would need to operate on a sub–tree of the membrane hierarchy and, hence, we will apply kind of similar auxiliary function to the previously computed root of the sub–tree.

The meaning of the rules in the definition is:

Axiom 1 describes how the translation function operates on the base case, where the node at issue is a leaf of the tree structure, that is a node with no children: in this case there is no recursive call; the function simply returns the looping sequence made of the parallel composition of the species onto the membrane and containing the parallel composition of the species into the membrane;

Rule 2 describes how the translation function operates in case of node with children: differently from the previous case, the function recursively calls itself on children's nodes; the result of these calls will be part of the parallel composition of terms contained in the looping sequence.

We formalize the initial term for a given MIM.

Definition 3.5 (Initial Term of a MIM, T_0)**.** *Given a Molecular Interaction Map, whose corresponding Tree Structure, describing the initial state of the model represented by the MIM, is* TS*, the initial term* T_0 *of the corresponding CLS+ model is defined as:*

$$T_0 \equiv [\![TS]\!]_T$$

Thus, in order to get the initial term of a given MIM, we need to follow two steps:

1. first, we define a Tree Structure representing a possible initial configuration of (labeled) elementary species;
2. the application of the translation function to this Tree Structure returns the corresponding initial CLS+ term.

Let's see how an initial term for map of Figure 3.5 is constructed. We have already defined TS, the Tree Structure representing the possible initial configuration of the model, as:

$$\begin{aligned} TS = \{C_0 &= \langle 0, \{1\}, \varnothing, \{(B,2)\}\rangle, \\ C_1 &= \langle 1, \{2\}, \{(A,3)\}, \varnothing\rangle, \\ C_2 &= \langle 2, \varnothing, \varnothing, \{(DNA,1)\}\rangle\} \end{aligned}$$

The initial term is then given by:

$$\begin{aligned} [\![TS]\!]_T = [\![C_0]\!] = &(0)^L \rfloor (B \mid B \mid [\![C_1]\!]) = \\ &(0)^L \rfloor (B \mid B \mid (1 \mid A \mid A \mid A)^L \rfloor ([\![C_2]\!])) = \\ &(0)^L \rfloor (B \mid B \mid (1 \mid A \mid A \mid A)^L \rfloor ((2)^L \rfloor (DNA))) \end{aligned}$$

3.2.2 Set of Rewrite Rules

We will now concentrate on a possible formalization for interaction symbols appearing in a MIM. We remark that our encoding will take into account only labeled reactions. We will examine different kind of reactions and we will give a formal method to produce one or more CLS+ rewrite rules for each labeled reaction in a map, modeling the expected behavior.

Interaction symbols include reaction and contingency symbol. The two kinds are strictly connected, since each reaction has a set of contingencies (which could be empty) affecting it. We already explained how reaction and contingencies will be interpreted in our formalization.

Formalization of Rewrite Rules

Given a set of species and a membrane hierarchy, we need a formal method to construct the CLS pattern which contains the given species and which leaves open the possibility of having third species, besides those strictly required.

We need an intermediate encoding for set of species, which could be reactants, products or contingencies. This representation must specify for each species its quantity and its position in the membrane hierarchy of the given map. Thus, we have to maintain a structure describing this membrane hierarchy. This structure is quite similar to the tree structure seen before, the only difference is that, this time a node only carries information about its children membranes.

The following is the formal definition of the membrane structure (a tree, indeed) describing the membrane hierarchy of a MIM.

Definition 3.6 (Membrane Structure of a MIM)**.** *Given a MIM, its* Membrane Structure MS *is a set of tuples of the form:*

$$\langle i, C\rangle \ \in \ \mathbb{N} \times 2^{\mathbb{N}}$$

where, each tuple, that is a node in the tree, corresponds to a membrane in the MIM and:

$i \in \mathbb{N}$ *is the membrane identification number; we assume $i = 0$ the id of the outmost membrane, which will be added to the MIM, in case it does not exist;*

$C \in 2^{\mathbb{N}}$ *is the set of identification numbers of the membranes which are immediately contained in the membrane at issue. If there are no such membranes, C will be equal to the empty set $\varnothing$;*

We denote with $\mathcal{MS}$ the set of all possible MIM Membrane Structures

For instance, the tree structure of the map of Figure 3.5 will be the following set of tuples:

$$MS = \{\langle 0, \{1\}\rangle, \langle 1, \{2\}\rangle, \langle 2, \varnothing\rangle\}$$

The following is the formal definition of the set of species, describing, for each species, its name, where they are displaced in the membrane hierarchy and their quantity.

Definition 3.7 (Set of Species)**.** *A* Set of Species SS *is a set of tuples of the form*

$$\langle S, (i, p), q\rangle \ \in \ \mathcal{S} \times (\mathbb{N} \times \{0, 1\}) \times \mathbb{N}$$

where, each tuple corresponds to a molecular species and:

$S \in \mathcal{S}$ *is the species name, that is the CLS+ sequence representing it;*

$(i, p) \in \mathbb{N} \times \{0, 1\}$ *is a pair representing the position in the membrane hierarchy, where the species at issue appears; i is the membrane id and $p = 0$ means that the species lies onto the membrane, while $p = 1$ means that the species is into the membrane;*

$q \in \mathbb{N}$ *is the amount of molecules of the species at issue.*

We denote with $\mathcal{SS}$ the set of all possible Sets of Species.

For instance, concerning MIM of figure 3.5, the set of species representing a possible initial situation, in which only elementary species exists, could be the following one:

$$\{\langle B,(0,1),2\rangle,\langle A,(1,0),3\rangle,\langle DNA,(2,1),1\rangle\}$$

Stochastic rewrite rules consist of a triple of two patterns and a rate constant. Informally, a reaction could be described by a rewrite rule in this way: the first pattern contains, in appropriate places, sequences representing the reactants and some term variable representing the eventual contingencies affecting the rate of this reaction and the rest of the surrounding environment; the second pattern is based on the first one: it maintains the variables representing the contingencies, it removes the reactants, which are consumed by the reaction, and it inserts the sequences corresponding to the products. We can deduce that each pattern can be traced out from the set of species which must appear in it, the first one can be traced out from contingency species and reactants, while the second one can be traced out from contingency species and products.

We now make a critical remark. We want rewrite rules to be as concise as possible: we do not want each rewrite rule specifying in its patterns the whole membrane structure explicitly, instead we want patterns to specify only the sub–context in which reactants, products and contingencies appears, leaving to the CLS+ semantics the task of, given the term modeling the whole map in a particular state, matching the rule patterns with this term. In other words, we would like to make the most from the following rules, belonging to the CLS+ semantics[4]:

$$\frac{T_1 \to T_2}{T \,|\, T_1 \to T \,|\, T_2} \qquad \frac{T_1 \to T_2}{\big(B\big)^L \rfloor T_1 \to \big(B\big)^L \rfloor T_2}$$

For instance, if species A and B appear in the exact same position $p = (i,1)$ and they can bind to form the complex $A{:}B$ (which will be at position p as well), we would like our semantics to specify only the strictly necessary, that is:

$$\big(\big(i \,|\, X_1\big)^L \rfloor (A \,|\, B \,|\, X_2), \big(i \,|\, X_1\big)^L \rfloor (A{:}B \,|\, X_2), k\big)$$

To this purpose we define a method to search the membrane structure for nodes (membranes), which are effectively necessary for the construction of the rule pattern. We anticipate that the set of necessary nodes forms a

[4]For sake of simplicity, we omit the stochastic details about rate constants, and we consider a rule as a pair of patterns

sub–tree in the original membrane structure: precisely, the minimum sub–tree containing all necessary membranes. A sub–tree will be combined with each reaction. We will give a formal method to compute such sub-tree, given a Membrane Structure and a Set of Species.

In a reaction we have three sets of species involved: reactants, products and contingencies. We observe, that, if some of these species appear in a certain membrane, then, for sure, this membranes is in the correspondent sub–tree. We define a function *membrane*, which, given a set of species, returns the set of those membranes (nodes in the membrane tree, that is natural numbers), which contain almeno at least one of the species in the given set. Formally:

Definition 3.8 (Set of membranes of a set of species, *membrane*)**.** *Given a set of species $SS \in \mathcal{SS}$, its set of membranes $membrane(SS) \subset Nat$ is defined as*

$$membrane(SS) = \{i \in \mathbb{N} | \langle i, p, q\rangle \in SS\}$$

Practically, $membrane(SS)$ is a set of nodes of a certain Membrane Structure.

Before giving the formal definition of the function computing the sub–tree containing some species, we define an auxiliary function *nodes*, which, given a membrane structure, returns the set of nodes appearing in it.

Definition 3.9 (Nodes in a Membrane Structure, *nodes*)**.** *Given a membrane structure MS, the set of nodes appearing in it, $nodes(MS)$, is defined as following:*

$$nodes(MS) = \{n \in \mathbb{N} | \langle n, C\rangle \in MS\}$$

Note that $nodes(\varnothing) = \varnothing$.

Now, we define the subtree *st* function, which, given a Membrane Structure of a MIM (which is indeed a tree rooted in 0) and a set of nodes, returns the minimum sub–tree containing all nodes in the given set of nodes.

Definition 3.10 (Sub–tree of a Membrane Structure containing a Set of Nodes, *st*)**.** *Given a membrane structure MS and a set of nodes N, having $\langle 0, C\rangle \in MS$, the function st, called on the pair composed by the tuple $\langle 0, C\rangle$ and N, returns a membrane structure MS', which is the minimum sub–tree* *of MS, containing all nodes in N. st is defined by the following rules:*

$$(base0)\quad \frac{n \in N}{st(\langle n, \varnothing\rangle, N) = \{\langle n, \varnothing\rangle\}}$$

$$(base1)\quad \frac{n \notin N}{st(\langle n, \varnothing\rangle, N) = \varnothing}$$

$$(end0)\quad \frac{cond \quad N \subseteq nodes(MS') \quad C' = necessaryChildren \quad |C'| > 1}{st(\langle n, \{c_1, c_2, \dots, c_k\}\rangle, N) = MS' \cup \{\langle n, nodes(C')\rangle\}}$$

$$(end1)\quad \frac{cond \quad N \subseteq nodes(MS') \quad C' = necessaryChildren \quad |C'| \leq 1}{st(\langle n, \{c_1, c_2, \dots, c_k\}\rangle, N) = MS'}$$

$$(mid0)\quad \frac{n \notin N \quad cond \quad C' = necessaryChildren = \varnothing}{st(\langle n, \{c_1, c_2, \dots, c_k\}\rangle, N) = \varnothing}$$

$$(mid1)\quad \frac{n \notin N \quad cond \quad C' = necessaryChildren \neq \varnothing}{st(\langle n, \{c_1, c_2, \dots, c_k\}\rangle, N) = MS' \cup \{\langle n, nodes(C')\rangle\}}$$

$$(mid2)\quad \frac{n \in N \quad cond \quad C' = necessaryChildren}{st(\langle n, \{c_1, c_2, \dots, c_k\}\rangle, N) = MS' \cup \{\langle n, nodes(C')\rangle\}}$$

where $|S|$ denotes the cardinality of set S. Condition "cond" mainly describes MS', which is the union of the membrane structures generated by recursive calls of st on the children of the node at issue. "cond" is equal to:

$$\begin{aligned} cond = \quad & \langle c_i, C_i\rangle \in MS \quad \wedge \quad MS_i = st(\langle c_i, C_i\rangle, N) \quad i = 1, 2, \dots, k \\ & \wedge \quad MS' = \bigcup_{j=1}^{k} MS_j \end{aligned}$$

$necessaryChildren$ describes C', which is the set of the tuple corresponding to the children of the node at issue that are also in MS': the motivation is that the children of each node, in the sub–tree of MS containing N, should not be the ones that are not in MS', that is, in constructing the new membrane structure, we discard the non–necessary children. $necessaryChildren$ is equal to:

$$necessaryChildren = \quad \bigcup_{j=1}^{k} \{\langle c_i, C_i\rangle\} \cap MS'$$

We briefly explain the meaning of each rule.

($base0$) states that the resulting membrane structure for a leaf n, which is in N is equal to the set containing the only tuple describing this leaf ($\langle n, \varnothing \rangle$);

($base1$) states that the resulting membrane structure for a leaf n, which is not N is equal to the empty set;

Rules ($end0$), ($end1$) describes the particular situation in which all the nodes in N are in the union of the sub–trees rooted in the children of the node at issue, that is $N \subseteq nodes(MS')$. When we reach this situation, we have almost finished in constructing the subtree of MS containing N. We distinguish two cases:

($end0$) is applied if $|C'| > 1$, that is, more than one sub–tree rooted in a child of the node at issue contains some node in N; in this case we must join all these sub–trees under a common root, that is the node at issue, to obtain a single tree: this tree will be the final sub–tree of MS containing N;

($end1$) is applied if $|F'| \leq 1$, that is, a single or zero sub–tree rooted in a child of the node at issue contains some node in N; in this case we have already found the final sub–tree of MS containing N, we just return it as it is.

Note that in these two cases, it is implicit that the node at issue is not in N.

Last we have the three rules, describing what happens in case st is called on a node with one or more children.

($mid0$) states that the sub–tree rooted at the node at issue n can be pruned, if none of its children is interested in the construction of the final membrane structure and n is not in N.

($mid1$) states that, if the node at issue n is not in N, but some of its children are interested in the construction of the final membrane structure, then a tuple containing n must anyway be in the final membrane structure, since there must be continuity in the resulting structure.

($mid2$) states that, if the node at issue n is in N, then a tuple containing it and its necessary children, must be inserted in the resulting membrane structure.

It might be that rule ($base0$) and ($base1$) are redundant with the last three rules, anyway, we keep them, since it seems to us that the whole definition is clearer.

We give some examples of st computations, given a certain set of nodes N.

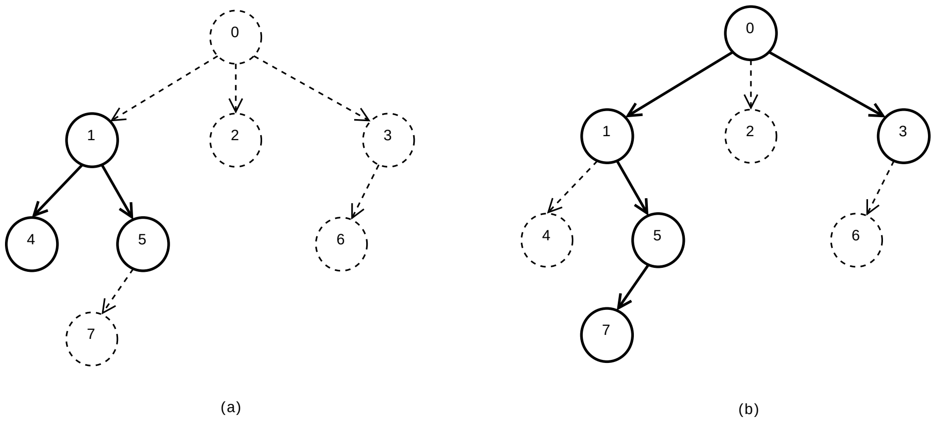

Figure 3.7 Example of Reaction Sub–trees for the membrane structure of Figure 3.1: (a) in evidence, sub–tree containing the set of nodes $N = \{4, 5\}$; (b) in evidence, sub–tree containing the set of nodes $N = \{1, 3, 7\}$

Suppose that we have a membrane structure of the only node $\langle 0, \varnothing \rangle$, that is only the external auxiliary membrane of a MIM. All species in a reaction will presumably appear in membrane 0, thus we have $N = \{0\}$. It is easy to verify that $st(\langle 0, \varnothing \rangle, \{0\}) = \{\langle 0, \varnothing \rangle\}$, resulting from the application of rule $base0$.

Now, suppose that 0 has two children $\langle 1, \varnothing \rangle$ and $\langle 2, \varnothing \rangle$. The tuple 0 is $\langle 0, \{1, 2\} \rangle$. Let $N = \{2\}$. We have:

$$\begin{aligned} st(\langle 1, \varnothing \rangle, N) &= \varnothing && (base1) \\ st(\langle 2, \varnothing \rangle, N) &= \{\langle 2, \varnothing \rangle\} && (base0) \\ st(\langle 0, \{1, 2\} \rangle, N) &= \{\langle 2, \varnothing \rangle\} && (end1) \end{aligned}$$

Thus, the eventual rewrite rule will have patterns containing only one looping, corresponding to membrane 2.

Now, please refer to Figure 3.1 and suppose we have $N = \{4, 5\}$, we expect st to compute the sub–tree shown in Figure 3.7.(a), in fact:

$$\begin{aligned} st(\langle 7, \varnothing \rangle, N) &= \varnothing && (base1) \\ st(\langle 5, \{7\} \rangle, N) &= \varnothing \cup \{\langle 5, \varnothing \rangle\} && (mid2) \\ st(\langle 4, \varnothing \rangle, N) &= \{\langle 4, \varnothing \rangle\} && (base0) \\ st(\langle 1, \{4, 5\} \rangle, N) &= \{\langle 4, \varnothing \rangle, \langle 5, \varnothing \rangle\} \cup \{\langle 1, \{4, 5\} \rangle\} && (end0) \\ st(\langle 2, \varnothing \rangle, N) &= \varnothing && (base1) \\ st(\langle 6, \varnothing \rangle, N) &= \varnothing && (base1) \\ st(\langle 3, \{6\} \rangle, N) &= \varnothing && (mid0) \\ st(\langle 0, \{1, 2, 3\} \rangle, N) &= \{\langle 4, \varnothing \rangle, \langle 5, \varnothing \rangle, \langle 1, \{4, 5\} \rangle\} \end{aligned}$$

$$\cup\, \varnothing \cup \varnothing \qquad (end1)$$

Now, please refer again to Figure 3.1 and this time suppose we have $N = 1, 3, 7$, we expect st to compute the sub–tree shown in Figure 3.7.(b), in fact:

$$\begin{aligned}
st(\langle 7, \varnothing\rangle, N) &= \{\langle 7, \varnothing\rangle\} && (base0)\\
st(\langle 5, \{7\}\rangle, N) &= \{\langle 7, \varnothing\rangle\} \cup \{\langle 5, \varnothing\rangle\} && (mid1)\\
st(\langle 4, \varnothing\rangle, N) &= \varnothing && (base1)\\
st(\langle 1, \{4, 5\}\rangle, N) &= \{\langle 7, \varnothing\rangle, \langle 5, \varnothing\rangle\} \cup \{< 1, \{5\}\rangle\} && (mid2)\\
st(\langle 2, \varnothing\rangle, N) &= \varnothing && (base1)\\
st(\langle 6, \varnothing\rangle, N) &= \varnothing && (base1)\\
st(\langle 3, \{6\}\rangle, N) &= \varnothing \cup \{\langle 3, \varnothing\rangle\} && (mid0)\\
st(\langle 0, \{1, 2, 3\}, N\rangle) &= \{\langle 4, \varnothing\rangle, \langle 5, \varnothing\rangle, \langle 1, \{5\}\rangle\}\\
&\quad \cup \{\langle 3, \varnothing\rangle\} \cup \{\langle 0, \{1, 3\}\rangle\} && (end0)
\end{aligned}$$

In the following, we need to know which node in a sub–tree of a Membrane Structure is the root. We define a function *root* that, given a Membrane Structure, returns the tuple corresponding to its root node. Note that if a breadth–first ordering is assumed, the root is the minimum node in the tree, and in a general case, the root is the only node with no father. We provide both definitions of *root*.

If we can assume a breadth–first ordering *root* of a Membrane Structure MS is defined as

Definition 3.11 (Root of a Membrane Structure [breadth–first ordering). *, $root(MS)$] Given a membrane structure in a breadth–first ordering of its node MS, $root : \mathcal{MS} \to (\mathbb{N} \times 2^{\mathbb{N}})$ returns the tuple of the root node appearing in it, and it is defined as following:*

$$root(MS) = min_n(MS)$$

where min_s, returns the tuple of the minimum node appearing in a Membrane Structure and is defined as following:

$$\frac{\langle \bar{i}, \overline{C}\rangle \in MS \quad (\forall \langle i, C\rangle \in MS \quad i \leq \bar{i})}{min_n(MS) = \langle \bar{i}, \overline{C}\rangle}$$

If we can not assume that nodes are numbered in breadth–first ordering, we could always define the root of a Membrane Structure as the unique node that has no father.

Definition 3.12 (Root of a Membrane Structure, $root(MS)$)**.** *Given a membrane structure MS, $root : \mathcal{MS} \rightarrow (\mathbb{N} \times 2^{\mathbb{N}})$ returns the tuple of the root node appearing in it, and it is defined as following:*

$$\frac{\langle \bar{i}, \overline{C} \rangle \in MS \quad (\forall \langle i, C \rangle \in MS \quad \bar{i} \notin C)}{root(MS) = \langle \bar{i}, \overline{C} \rangle}$$

Now, we provide an intermediate encoding for contingencies and reactions. Each reaction is combined with a set of contingencies, which contains pairs of a set of species and a rate constant. Each pair describes the kind and amount of species that must exist for the contingency to be verified and the rate constant at which the reaction can happen, if the contingency at issue is verified. For each reaction, we have to compute the sub–tree containing the species appearing in the reaction: either contingencies, reactants or products.

We define the set of contingencies as following:

Definition 3.13 (Set of Contingencies, CS)**.** *A* Set of Contingencies *RS is a set of pairs of the form*

$$\langle C, k \rangle \ \in \ \mathcal{SS} \times \mathbb{R}$$

where, each tuple corresponds to a contingency symbol and:

$C \in \mathcal{SS}$ *is the set of species that must exist for the contingency to be verified;*

$k \in \mathbb{R}$ *is the reaction rate constant, at which a certain reaction can happen if this contingencies are verified.*

We denote with $\mathcal{CS}$ the set of all possible Sets of Contingencies.

In mathematical terms, the mapping of each contingency to a certain rate constant, can be seen as a function that, given a set of species, returns the respective rate. We should note, that practically, for sake of simplicity, we assume that in designing the map, a designer does not use the same rate constant for two contingencies: if two contingencies will have the same rate constant, then they should be combined with some state–combination symbol or some other notation. In alternative, we could have used a unique identifier for each contingency, however, we preferred not to make the representation heavier. Thus, we assume that contingencies affecting the same reaction can be identified by their rate constant.

For instance, the two Sets of Contingencies for the two reactions (binding and unbinding) appearing in the map of Figure 3.2.(a) are the following ones:

$$C_1 = \{\langle \varnothing, k1 \rangle, \langle \{\langle C, (0,1), 1 \rangle\}, k1' \rangle, \langle \{\langle C, (0,1), 1 \rangle, \langle D, (0,1), 1 \rangle\}, k1" \rangle\}$$

$$C_2 = \{\langle \varnothing, k2 \rangle, \langle \{\langle C, (0,1), 1 \rangle\}, k2' \rangle, \langle \{\langle C, (0,1), 1 \rangle, \langle D, (0,1), 1 \rangle\}, k2" \rangle\}$$

Note that here we assumed that one instance of each contingency species is enough for the contingency to be verified.

As previously said, a reaction symbol is described by its reactants and products combined with a set of contingencies plus the Membrane Structure representing the sub–tree involved in the reaction. We now give the formal definition for the structure representing the set of reactions in a map, the so called *Set of Reactions RS*.

Definition 3.14 (Set of Reactions, RS)**.** *Being MS the Membrane Structure of a given MIM, having $n = \langle 0, C\rangle \in MS$, its* Set of Reactions *RS is a set of tuple of the form*

$$\langle CS, R, P, MS'\rangle \ \in \ \mathcal{CS} \times \mathcal{SS} \times \mathcal{SS} \times \mathcal{MS}$$

where, each tuple corresponds to a reaction symbol in a MIM and MS' is defined as following:

$$\frac{CS = \{(C_1, k_1), (C_2, k_2), \ldots, (C_m, k_m)\} \quad M = membrane\left(\left(\bigcup_{j=1}^{m} C_j\right) \cup R \cup P\right)}{MS' = st(n, M)}$$

The meaning of each element in the tuple is the following:

$CS \in \mathcal{CS}$ is the set of contingencies affecting the reaction at issue;

$R \in \mathcal{SS}$ is the set of reactants;

$P \in \mathcal{SS}$ is the set of products;

$MS' \in \mathcal{MS}$ is the sub–Membrane Structure containing all the species in this reaction.

We denote with $\mathcal{RS}$ the set of all possible Sets of Reactions.

Note that, if a reaction has no contingencies acting on it, its set of contingencies will contain the only pair $(\varnothing, k)$.

For instance, the Set of Reactions appearing in the map of Figure 3.2.(a) is the following:

$$\begin{aligned} RS = \{R_1 =& \langle C_1, \{\langle A, (0,1), 1\rangle, \langle B, (0,1), 1\rangle\}, \{\langle A{:}B, (0,1), 1\rangle\}, MS\rangle, \\ R_2 =& \langle C_2, \{\langle A{:}B, (0,1), 1\rangle\}, \{\langle A, (0,1), 1\rangle, \langle B, (0,1), 1\rangle\}, MS\rangle\} \end{aligned}$$

where the two set of contingencies C_1 and C_2 are as previously computed and $MS = \{\langle 0, \varnothing\rangle\}$.

We are at a crucial point in our translation. We are going to define a method that, given a Membrane Structure, a Set of Contingencies and two

Set of Species, constructs two CLS+ patterns, whose looping sequences corresponds to the membrane structure and which contains the respective given species at their exact place. The patterns will have the same term variables representing the possible presence of third species or contingency species at each level in the membrane structure. We call this method *Parallel Pattern Builder*. As the reader probably intuited, the Parallel Pattern Builder will be used in the construction of rewrite rules: the two sets of species being reactants and products; reactants will appear as sequences in the first pattern, products as sequences in the second one.

The delicate point is how the two patterns keep track of contingencies: term variables will implicitly model possible contingencies. This term variables must be equal in both patterns, that is contingency species are not modified by a possible reaction: this gives the motivation for building the two patterns "in parallel", since they "share" some variable. Since contingency species can be displaced over different membranes, there can be more than one term variable modeling them. Each term variable has a set of species that could appear in it. For each contingency of each reaction, we must keep track of which set of contingency species is contained in each term variable: this information gathering is done during the procedure *Parallel Pattern Builder*, hence, this procedure returns a triple of two patterns and a structure that keeps this information about contingencies.

We first define a translation function that, given a set of species, produces the parallel composition of the terms representing each species, taking into account the quantity of molecules of each species. This function is quite similar to the ϕ previously defined, thus we call it ϕ_s.

Note that, in the Parallel Pattern Builder, this function will be applied only to set of species belonging to the exact same place in the membrane structure (such a place is identified by the pair $(membrane_{id}, onto/into)$).

Recall that $\mathcal{T}$ denotes the infinite set of possible terms and $\mathcal{P}$ denotes the infinite set of possible patterns.

Definition 3.15 (Set of Species Translation Function, ϕ_s)**.** *Given a set of species SS, the function $\phi_s : \mathcal{SS} \to \mathcal{T}$ constructs the correspondent CLS+ parallel composition $T = \phi_s(SS)$, according to the following definition:*

$$\phi_s(\varnothing) = \epsilon$$

$$\frac{S_i = \langle s_i, p_i, n_i \rangle \qquad i = 1, 2, \ldots, q}{\phi_s\{S_1, S_2, \ldots, S_q\} = n_1 \oplus s_1 \,|\, n_2 \oplus s_2 \,|\, \ldots \,|\, n_q \oplus s_q}$$

where s_i are generic CLS+ Sequences, $n \oplus T$ stands for a parallel composition $T \,|\, T \,|\, \ldots \,|\, T$ of length n (that is, T appears n times) and T is a generic CLS+ Term. Note that $0 \oplus T \equiv \epsilon$.

Notice that ϕ_s applied to an empty set of species returns the empty sequence.

In the definition of Parallel Pattern Builder, we use a structure to memorize which term is assigned to each contingency. We call this structure *Set of Contingencies Star* $CS^\star$, since it has some more information than a Set of Contingencies.

Definition 3.16 (Set of Contingencies Star, $CS^\star$)**.** *A* Set of Contingencies *RS is a set of pairs of the form*

$$\langle XC, k\rangle \ \in \ \mathcal{SS} \times \mathbb{R}$$

where, each tuple corresponds to a contingency symbol and:

$XC \in 2^{(TV \times \mathcal{SS})}$ *is a set of pairs of the kind $\langle X, SS\rangle$, where X is a Term Variable and SS is the set of contingency species which, in the rewrite rule patterns, could appear in the context of X;*

$k \in \mathbb{R}$ *is the reaction rate constant, at which a certain reaction can happen if this contingencies are verified.*

We denote with $\mathcal{CS}^\star$ the set of all possible Sets of Contingencies Star.

In order to be able to recursively manage all the Sets of Contingencies Star created by the children of a given node in the definition of the Parallel Pattern Builder, we need a function that merges such sets to a unique one in the following way:

Definition 3.17 (Merge of Sets of Contingencies Star, *merge*)**.** *Given n Sets of Contingencies Star $CS^\star_1, CS^\star_2, \ldots, CS^\star_n$, the function merge returns a single Set of Contingencies Star $CS^\star$ defined in the following way:*

$$merge(CS^\star_1, CS^\star_2, \ldots, CS^\star_n) = \\ \{\langle XC, k\rangle | \langle XC_i, k\rangle \in CS^\star_i \quad union(XC_1, XC_2, \ldots, XC_n)\}$$

where

$$union(XC_1, XC_2, \ldots, XC_n) = \\ \{\langle X, SS\rangle | \langle X, SS_i\rangle \in XC_i \quad SS = \bigcup_{i=1}^{n} SS_i \quad i = 1, 2, \ldots, n\}$$

We can note how this definition works only because we assumed the rate constant being a unique identifier for each contingency in a set of contingencies.

We remark that, since a reaction operates only on a sub–tree of the membrane structure of a MIM, the Parallel Pattern Builder should start from the root of such a tree, which is computed using the proper *root* function definition.

Definition 3.18 (Parallel Pattern Builder, $[\![\cdot]\!]_{pp}$). *Given a membrane structure MS and a Set of Contingencies CS and two Sets of Species SS_1, SS_2, having $n = root(MS)$, the* parallel pattern builder $[\![\cdot]\!]_{pp} : \mathcal{MS} \times (\mathcal{CS} \times \mathcal{SS} \times \mathcal{SS}) \to (\mathcal{CS}^\star \times \mathcal{P} \times \mathcal{P})$ *is defined as following:*

$$[\![MS, (CS, SS_1, SS_2)]\!]_{pp} = [\![n, (CS, SS_1, SS_2)]\!]_p$$

where function $[\![\cdot]\!]_p$, taking a tuple in the membrane structure and two sets of species and returning a Set o Contingencies Star and pair of patterns, is defined by the following rules:

$$\frac{\begin{array}{c} X_0, X_1 = new(TV) \\ CS^\star = \{\langle\{\langle X_0, C^{i0}\rangle, \langle X_1, C^{i1}\rangle\}, k\rangle \;|\langle C, k\rangle \in CS\} \\ \phi_s(SS_1^{i0}) = T1_{i0} \quad \phi_s(SS_1^{i1}) = T1_{i1} \qquad \phi_s(SS_2^{i0}) = T2_{i0} \quad \phi_s(SS_2^{i1}) = T2_{i1} \end{array}}{\begin{array}{l} [\![\langle i, \varnothing\rangle, (CS, SS_1, SS_2)]\!]_p = (CS^\star, \\ \qquad \big(i \,|\, T1_{i0} \,|\, X_0\big)^L \rfloor (T1_{i1} \,|\, P1_1 \,|\, P1_2 \,|\, \ldots \,|\, P1_k \,|\, X_1), \\ \qquad \big(i \,|\, T2_{i0} \,|\, X_0\big)^L \rfloor (T2_{i1} \,|\, P2_1 \,|\, P2_2 \,|\, \ldots \,|\, P2_k \,|\, X_1)) \end{array}}$$

$$\frac{\begin{array}{c} X_0, X_1 = new(TV) \\ \langle c_j, C_j\rangle \in MS \\ (CS_j^\star, P1_j, P2_j) = [\![\langle c_j, C_j\rangle, (SS_1, SS_2)]\!]_p \qquad j = 1, 2, \ldots, k \\ CS^\star = \{\langle\{\langle X_0, C^{i0}\rangle, \langle X_1, C^{i1}\rangle\}, k\rangle \;|\langle k, C\rangle \in CS\} \\ \overline{CS^\star} = merge(CS^\star, CS_1^\star, CS_2^\star, \ldots, CS_k^\star) \\ \phi_s(SS_1^{i0}) = T1_{i0} \quad \phi_s(SS_1^{i1}) = T1_{i1} \qquad \phi_s(SS_2^{i0}) = T2_{i0} \quad \phi_s(SS_2^{i1}) = T2_{i1} \end{array}}{\begin{array}{l} [\![\langle i, \{c_1, c_2, \ldots c_k\}\rangle, (CS, SS_1, SS_2)]\!]_p = (\overline{CS^\star}, \\ \qquad \big(i \,|\, T1_{i0} \,|\, X_0\big)^L \rfloor (T1_{i1} \,|\, P1_1 \,|\, P1_2 \,|\, \ldots \,|\, P1_k \,|\, X_1), \\ \qquad \big(i \,|\, T2_{i0} \,|\, X_0\big)^L \rfloor (T2_{i1} \,|\, P2_1 \,|\, P2_2 \,|\, \ldots \,|\, P2_k \,|\, X_1)) \end{array}}$$

where $SS^{ij} \subset SS$ denotes the subset of species which position is equal to (i, j), that is

$$SS^{ij} = \{s \in SS | s = \langle S, (i, j), q\rangle\}$$

where S is a general species name and q is a natural number; $X_i = new(TV)$ means that the term variable X_i has not previously used in the current application of $[\![\cdot]\!]_{pp}$; P_i are general patterns, T_i are general terms.

The first rule operates on leafs, while the second one operates on nodes with children. We observe that, in practice, in merging two Sets of Contingencies Star, we could use the usual union on sets, instead of a special *union* function, since the term variables used in the recursive calls of $[\![\cdot]\!]_p$ are different.

As we have seen in section 3.1, in the end, to each rewrite rule is combined the definition of its rate function. This definition depends on the Set of Contingencies Star combined with each rewrite rule. We proceed with the definition of the translation function that given the Set of Reactions of a map, produces a set of stochastic rewrite rules, termed *Set of Reactions Translation Function* $[\![\cdot]\!]_{rr}$ The following is its definition. We denote with $\Re$ the infinite set of possible stochastic CLS+ rewrite rules.

Definition 3.19 (Set of Reactions Translation Function, $[\![\cdot]\!]_{rr}$). *Given a set of n reactions $RS = \{R_1, R_2, \ldots R_n\}$, the* Set of Reactions Translation Function $[\![\cdot]\!]_{rr} : \mathcal{RS} \to \Re$ *is defined as following:*

$$[\![RS]\!]_{rr} = \{[\![R_1]\!]_r, [\![R_2]\!]_r, \ldots, [\![R_n]\!]_r\}$$

where function $[\![\cdot]\!]_r$, which takes a single reaction tuple, is defined by the following rule:

$$\frac{(CS^\star, P1, P2) = [\![root(MS), (CS, RS, PS)]\!]_{pp}}{[\![\langle CS, RS, PS, MS\rangle]\!]_r = (P1, P2, f)}$$

where $P1_i, P2_i$ are patterns and, having

$$CS^\star = \{\langle k, \varnothing\rangle, \langle k_1, XC_1\rangle, \langle k_2, XC_2\rangle, \ldots, \langle k_n, XC_n\rangle\}$$

and

$$XC_i = \{\langle X_1, S_1\rangle, \langle X_2, S_2\rangle, \ldots, \langle X_{m_i}, S_{m_i}\rangle\}$$

the rate function f is defined as following:

$$f = \begin{cases} k_1 & \text{if } \sigma(X_1) \equiv \phi_s(S_1)|T_1 \wedge \sigma(X_2) \equiv \phi_s(S_2)|T_2 \wedge \sigma(X_{m_1}) \equiv \phi_s(S_{m_1})|T_{m_1} \\ k_2 & \text{if } \sigma(X_1) \equiv \phi_s(S_1)|T_1 \wedge \sigma(X_2) \equiv \phi_s(S_2)|T_2 \wedge \sigma(X_{m_2}) \equiv \phi_s(S_{m_2})|T_{m_2} \\ \vdots & \vdots \\ k_n & \text{if } \sigma(X_1) \equiv \phi_s(S_1)|T_1 \wedge \sigma(X_2) \equiv \phi_s(S_2)|T_2 \wedge \sigma(X_{m_n}) \equiv \phi_s(S_{m_n})|T_{m_n} \\ k & \text{otherwise} \end{cases}$$

Note that this semantics does not produce CLS+ Brane Rules, since, due to the need of making each reaction local to a precise sub–membrane structure, each rule will always consist of at least a looping sequence.

As a careful reader could have noticed, differently from the specification of the rate functions of the examples in section 3.1 (Figure 3.2), here we did not worry about making sure that if a certain contingency is verified, there are no other "more specific" contingencies verified, which should overcome that contingency. Saying that a certain contingency $c_1 = \langle C_1, k_1\rangle$ is "more specific" then another one $c_2 = \langle C_2, k_k\rangle$, we intend that contingency species of c_1 are all the ones of c_2 plus some other species, that is $C_2 \subset C_1$. For instance, in the example of Figure 3.2, the rate constant k_1' for complexation shall be applied only if there is no instance of species D;

in our formalization, we assume that, since rate k_1" refers to a more specific contingency it shall be preferred to k'_1, when both contingencies are verified. We could also write the conditions in the rate function in a decreasing partial[5] order of specificity, stating that the first term conditions matching the variable instantiations returns the rate constant. Thus, when more than one contingency is verified, we assume the rate function to return the rate constant to be the one of the most specific verified contingency.

We remark again how contingencies could be formalized in a more accurate way: we could easily define a function that computes the concentration of a certain species in a given parallel composition of sequences. We could then use a rate function depending on the concentrations of contingency species.

We formalize the Set of Rewrite Rules for a given MIM.

Definition 3.20 (Set of Rewrite Rules of a MIM, RR). *Given a Molecular Interaction Map, whose corresponding Membrane Structure is MS and whose reactions are encoded in the Set of Reactions RS, the Set of Rewrite Rules $RR \subset \Re$ of the corresponding Stochastic CLS+ model is defined as:*

$$RR = [\![RS]\!]_{rr}$$

Thus, in order to construct the Set of Rewrite Rules of a given MIM, we need to follow two steps:

1. we encode the labeled reactions in a Set of Reactions;
2. the application of the translation function to this Set of Reactions returns the corresponding Set of Rewrite Rules.

Considerations on semantics for Reaction Symbols

After having bombed the reader with a cascade of definitions, let's see how reactions appearing in a MIM could be formalized in practice.

Take as example the MIM of Figure 3.8, where we added rate constants to each interaction. Rate constants paired with contingency symbols, denote the rate constant at which the pointed reaction can happen, when the contingency is satisfied. As we have seen before, its membrane structure is:

$$MS = \{\langle 0, \{1\}\rangle, \langle 1, \{2\}\rangle, \langle 2, \varnothing\rangle\}$$

[5]contingencies having disjoint sets of contingency species can not be compared, they simply refer to non–interfering situations

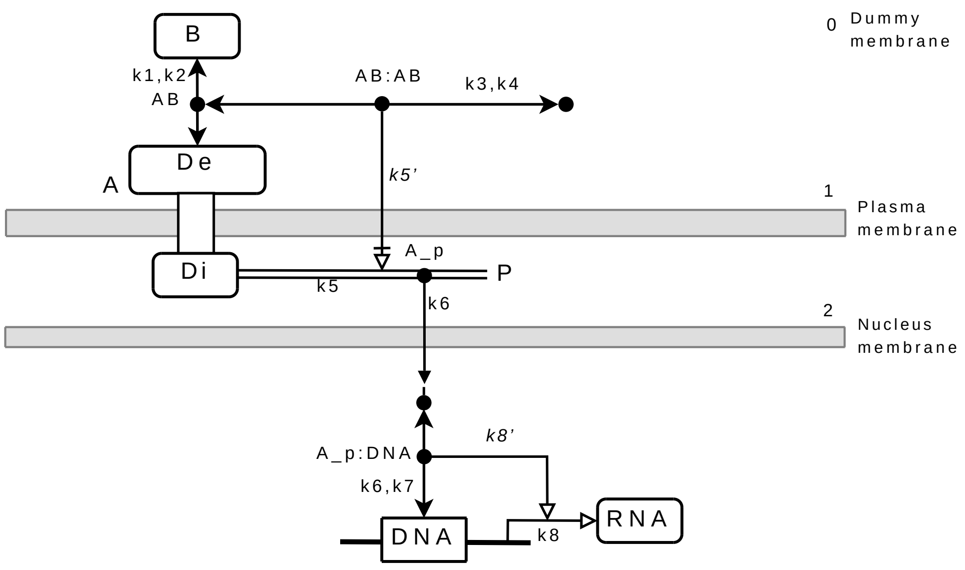

Figure 3.8 Example of named Explicit MIM diagram; rate constants are paired to each reaction and contingency: rates of contingencies are written in *italic*, since, in practice, they "substitute" the original rate constants of the pointed reaction, when the contingency is satisfied.

We want to formalize the reaction:

$$A + B \underset{k_2}{\overset{k_1}{\rightleftharpoons}} AB$$

Note that we described a "1–to–1" reaction, where a molecule of A and a molecule of B can be complexated into a molecule of $A\!:\!B$. This **non–covalent binding** is quite simple, since it is not affected by any contingency. Indeed, the reaction shows a reversible binding, which represents two reactions, one for complexation with rate k_1 and the respective decomplexation, with rate k_2.

Thus, we need to describe two reaction tuples for this reaction:

$$R_1 = \langle\{\langle\varnothing, k_1\rangle\}, \{\langle A, (1,0), 1\rangle, \langle B, (0,1), 1\rangle\}, \{\langle AB, (1,0), 1\rangle\}\}, MS'\rangle$$

$$R_2 = \langle\{\langle\varnothing, k_2\rangle\}, \{\langle AB, (1,0), 1\rangle\}, \{\langle A, (1,0), 1\rangle, \langle B, (0,1), 1\rangle\}\}, MS'\rangle$$

where MS' is the sub–tree containing all the species in the reaction and it is equal to $MS = \{\langle 0, \{1\}\rangle, \langle 1, \varnothing\rangle\}$

We reasonably place the product on membrane 1, since the complex should be still connected to the membrane. The resulting rewrite rules for these reactions are:

$$\begin{aligned}(R_1)\quad & (\big(0\,|\,X_0\big)^L \rfloor (B\,|\,\big(1\,|\,A\,|\,X_2\big)^L \rfloor X_3\,|\,X_1),\\ & \big(0\,|\,X_0\big)^L \rfloor (\big(1\,|\,AB\,|\,X_2\big)^L \rfloor X_3\,|\,X_1),\\ & k_1)\end{aligned}$$

$$\begin{aligned}(R_2)\quad & (\big(0\,|\,X_0\big)^L \rfloor (\big(1\,|\,AB\,|\,X_2\big)^L \rfloor X_3\,|\,X_1),\\ & \big((0\,|\,X_0) \rfloor (B\,|\,\big(1\,|\,A\,|\,X_2\big)^L \rfloor X_3\,|\,X_1)\big)^L,\\ & k_2)\end{aligned}$$

where we used directly the rate constant, since, indeed, the rate function would have been a constant.

The description of a **covalent binding** is essentially the same, with the difference that only it denotes only a possible complexation, thus, there is only one rewrite rule modeling it. The converse rule for decomplexation is possible only if a **cleavage** symbol acts on the covalent binding. In this case, a rule similar to what we did for the decomplexation of a reversible binding is constructed, with the only difference in the eventual presence of a third species acting as "bond–breaker": this last species could be seen as a contingency necessary for the decomplexation to happen. **Translocation** can be easily represented with this system, just imagine a reaction tuple which contains in its reactants the species in its original position and in its products the same species in a different position. Let's go on with our parade of reactions. The representation of **stoichiometric conversion**, **lossless production**, **transcription** and **degradation** as a reaction tuple and their conversion into rewrite rules are really intuitive and the reader has probably already guessed all of them. Species A stoichiometrically converting to species B is represented by a reaction tuple having A as the only reactant and B as the only product. Lossless production is very similar to the stoichiometric conversion: the only difference is that the reactant (or reactants) also appears (appear) in the set of products, since it is (they are) not consumed by the reaction. Transcription is basically a lossless production involving some DNA molecule as reactant. The degradation of some species A corresponds to a reaction tuple which has A in its set of reactants and the empty set as its products.

Reaction **in–trans**, that is between two different molecules belonging to the same species can also be represented with a reaction tuple. The tuple will specify that two different instances of the same species are reacting together, by simply having as many copies of the species as are needed in the set of reactants species.

Note how this notation could flexibly represent loads of different reactions: for instance, it could be possible to represent many translocations all–in–one, with a single rewrite rule; we could theoretically represent simultaneous reactions, which are independent one from each other and could hence be "executed at the same time". Moreover, it is possible to specify for each reaction the amount of molecules necessary for each reactant and product (stoichiometric coefficients) or even the amount of some contingency species, which could be necessary for the reaction to happen.

This freedom in expressing reactions, could sometime result in not realistic reactions. For instance, one could describe a reaction between species that belongs to different membranes: it is quite unrealistic to believe that two molecules, separated by one or more membranes, can stick together. Tracing reasonable reactions is left up to the designer. A designer is theoretically free to trace interaction lines traversing multiple stratus of membranes: such interactions will be encoded in the CLS+ model. This freedom faithfully represents the freedom of MIM notation, which do not formally state any limitation due to membranes; however, in our experience, we have never encountered MIMs with fancy interactions between species that are not in contact with each other.

The formalization of a map results indeed fairly easy: we modeled all different symbols appearing in a map with one single structure: the set of reactions.

What played in our favor in this formalization, is that the explicit interpretation allows us to name all possible existing species in advance. Thus, we are not forced to look at interactions at domain level and it does not make a big difference if the species can have an intra–molecular binding, since this will be simply denoted by a species with a new name.

Considerations on Semantics of Contingency Symbols

This time we show as example, how a rewrite rule for the phosphorylation of **A** in the map of Figure 3.8 can be constructed. In the explicit interpretation of this map, the phosphorylation can happen only if the homodimer $AB{:}AB$ has been formed. We remark that, the explicit interpretation of this map is not intended to make sense from a biological point–of–view, however it can be used as example of contingency. As usual the membrane structure is:

$$MS = \{\langle 0, \{1\}\rangle, \langle 1, \{2\}\rangle, \langle 2, \varnothing\rangle\}$$

A **modification** will be described with a single reaction tuple, since, in practice it is a covalent binding. The presence of a contingency pointing to the reaction will be a tuple in the Set of Contingencies. Thus, we will formalize the following reactions:

$$(a) \qquad A \overset{k_5}{\rightleftharpoons} pA$$

$$(b) \qquad AB{:}AB + A \overset{k_5'}{\rightleftharpoons} AB{:}AB + pA$$

A single reaction tuple will model both reactions:

$$R = \langle \{ \langle \varnothing, k_5 \rangle, \langle \{ \langle AB{:}AB, (0,1), 1 \rangle \}, k_5' \rangle \}, \\ \{ \langle A, (1,0), 1 \rangle \}, \{ \langle pA, (1,0), 1 \rangle \} \}, MS' \rangle$$

where, k_5 could be equal to zero and $MS' = \{ \langle 0, \{1\} \rangle, \langle 1, \varnothing \rangle \}$

Note that, since phosphate **P** is one of the so–called ubiquitous molecules, it is not modeled as a species, since in general, there are thousands of such particles dispersed all over a cell. We reasonably place the product pA on membrane 1, since the complex should be still connected to the membrane. The resulting rewrite rule for this reaction tuple is:

$$\begin{aligned}(a) \quad & \big((0 \mid X_0)^L \rfloor \big((1 \mid A \mid X_2)^L \rfloor X_3 \mid X_1\big), \\ & (0 \mid X_0)^L \rfloor \big((1 \mid pA \mid X_2)^L \rfloor X_3 \mid X_1\big), \\ & f\big)\end{aligned}$$

$$f = \begin{cases} k_5' & \text{if } \sigma(X_1) \equiv AB{:}AB|T \\ k_5 & \text{otherwise} \end{cases}$$

In a similar way we can model all kind of contingencies. Catalysis, stimulation, requirement and inhibition are all modeled as tuning the reaction rate to an appropriate one.

As for reactions, we can note the freedom in expressing contingencies: a certain species can possibly affect each of the reactions appearing in the map; there is not any obligation of having contingency species and reactants or products in the same membrane. Depicting meaningful contingencies is left up to the designer.

3.2.3 Signal Transduction: the EGF Signalling Pathway

At the cellular level, signal transduction refers to any process by which a signal moves from outside the cell to inside, mediating the sensing and processing of stimuli by the cell. Cells are highly responsive to signals sent by molecules in its environment. Only few molecules are able to pass through the cell membranes and, once inside the cell, bind to proteins that interact directly with DNA and modulate gene transcription. Most of the times, signal molecules are too large to cross the membrane. Thus, the signal of this molecules must be transmitted across the cell membrane without the molecules themselves entering the cell. In order to recognize that a certain signalling molecule is present in the environment, cells offer some specific transmembrane receptor protein, which transfer the signal across the membrane. The receptor has an extracellular domain and an intracellular domain. The extracellular domain can bind to a signal protein. After such binding is established, the receptor undergoes some conformational change which activates its intracellular domain. The activation of this domain causes a cascade of interactions with other molecules inside the cell, which may produce different effects on the cell. This sequence of interactions is usually termed *signalling pathway.*

An example of such signal transduction is the Epidermal Growth Factor (EGF) signalling pathway, which regulates cell growth, proliferation, and differentiation. The epidermal growth factor receptors (EGFR) are located on the cell surface and they are activated by the binding of their extracellular domain with a EGF, forming a EGFR (ligand–receptor) complex. This causes a conformational change of the so activated EGFR, which enables it to bind to another activated EGFR to form a dimer. EGFR dimerization stimulates the phosphorylation of its intracellular domain, which, in its turn, activates a signalling proteins named ShC by binding to it. Activated ShC initiate a cascade of signal transductions, which in the end, lead to some DNA transcription and hence cell proliferation.

Kohn *et al.* in [11, 12] show an explicit MIM diagram for the EGFR pathway. Figure 3.9 shows the entire diagram as it appears in [11]. As we remarked in section 1.2.4, official explicit MIMs utilize only a subset of MIM symbols: reaction symbols and the catalysis symbol and this last is interpreted as reactions. Thus, there are not real contingencies in this map.

We would like to model in CLS+ the first steps of this signal pathway, precisely until reaction 10, where Ras:GTP converts to Ras:GTP, that is, we ignore species greater than 16 and reactions greater than 10. We here report the supplementary description for these reactions as found on [11], for further information on other reactions please refer to the cited article.

"EGF, an extracellular growth factor, binds the extracellular (receptor) domain of EGFR (interaction–1). The EGFR molecule (species–2) is shown as composed of extracellular, transmembrane, and intracellular parts. The EGF:EGFR complex (species–3) then forms a homodimer (interaction–2, species–4), which autophosphorylates several tyrosine sites in the intracellular part of the EGFR molecules (interaction–3). Interaction–3 is a stoichiometric conversion of the EGF:EGFR dimer to an EGF:phosphorylated–EGFR dimer (species–5) (the node on the arrowless lines represents phosphorylated–EGFR (state–combination symbol is explained in section 1.2.2, A.N.). *Two of these phosphotyrosine sites (Y1068 and Y1086)*

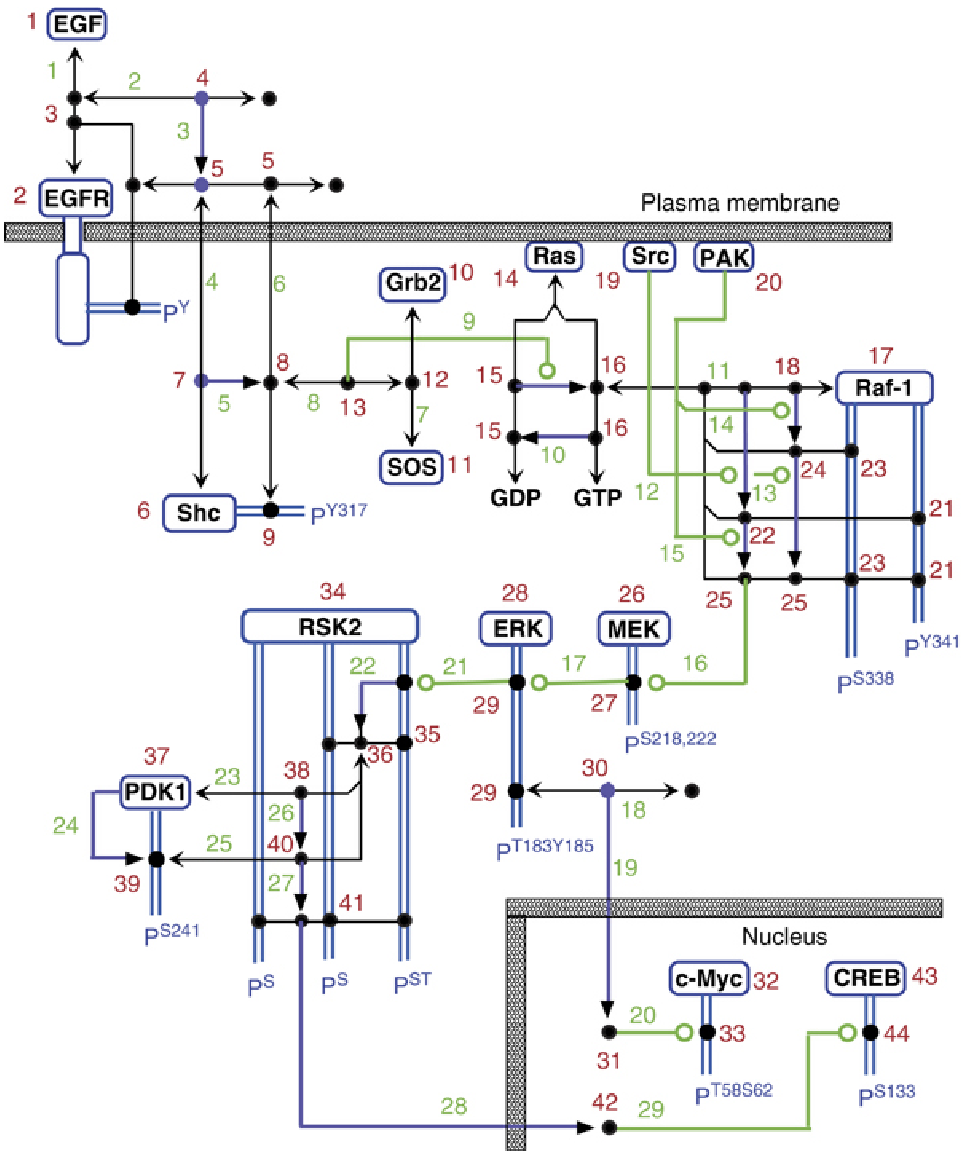

Figure 3.9 Explicit MIM of signaling from the epidermal growth factor receptor (EGFR, ErbB1). Molecular species are numbered in red and reactions are numbered in green italics. A detailed description of the MIM can be found at http://discover.nci.nih.gov.

(species–5) bind Shc (interaction–4). (Species–5 is a dimer of the EGF:EGFR complex. Note that two nodes placed within the same interaction line refer to the same molecular species.) The bound Shc (species–7) is then phosphorylated at Y317 (species–9) by the tyrosine kinase domain on the intracellular part of the EGFR molecule; in effect, this converts the EGFR:unphosphorylated–Shc complex (species–7) to EGFR:phosphorylated–Shc (interaction–5, species–8). EGFR and phosphorylated–Shc bind to each other reversibly (interaction–6). (The release of phosphorylated–Shc into the cytosol was not included in the original process diagram. This example shows how MIM diagrams can suggest actions that might not otherwise have been considered.) The Grb2:SOS complex (interaction–7, species–12) binds to the phosphorylated–EGFR:phosphorylated–Shc complex (interaction – 8, species–13), and is thereby recruited to the plasma membrane. This places the SOS into position where it can act on membrane–bound Ras (species–14). The SOS component of the Grb2:SOS complex then binds to Ras:GDP (species–15) and enzymatically converts Ras:GDP to Ras:GTP (species–16, interaction–9). (Note that interaction–9 is enzymatic and therefore has three component reactions, labeled 9a, 9b, and 9c in Table1.) Ras has a GTPase function that slowly converts itself from the Ras:GTP to the Ras:GDP form (interaction–10). (There are 2 nodes within the Ras:GDP line referring to species–15, and 2 nodes within the Ras:GTP line referring to species–16.)"

We report the part of our interest of the reaction table describing the reactions in the explicit EGFR map of Figure 3.9 (for the full table please refer to [11]). We added a fourth column for the kinetic constant.

In order to be able to translate the model into CLS+, first, we need to encode the part of the map of our concern into the intermediate representation. We assign a CLS+ name to possible species (labeled species). We should remark how, phosphorylated–EGFR species is not given a number: indeed, the covalent reaction leading to this species does not happen in practice; this fact could be represented by giving the reaction a rate constant equal to zero or by simply ignoring the reaction while building the model. As previously mentioned, we adopt this second strategy and thus, we do not insert such reactions into the intermediate representation. Further "dummy" reaction, which are used only to represent a complex state, are: the non–covalent binding yielding species 5, that is the dimer of two EGF:phosphorylated–EGFR; the non–covalent binding between Ras and GDP, yielding Ras:GDP; and the non–covalent binding between Ras and GTP, yielding Ras:GTP. Thus, in practice a name is given only to those species which are numbered in the diagram. Table 3.2 lists name and a possible initial concentration of each of the seventeen species involved in the reactions of interest (note that, in practice, species 14 (Ras) is not present in its elementary form, since the non–covalent reactions with GDP or GTP are not labeled).

Table 3.1 Part of the reaction table of the reactions in the explicit MIM shown in Figure 3.9. "*The numbers refer to the reaction and species identification numbers in Figure 3.9. For reversible binding, the letter "a" or "b" is appended to refer to association and dissociation, respectively. For enzyme reactions, suffixes "a" and "b" refers to production and dissociation of the enzyme–substrate complex, respectively; suffix "c" refers to conversion of the enzyme–substrate complex to products. A letter suffix added to a reactant or product species refers to the enzyme–substrate complex (whose existence is implied by the enzyme reaction symbol in the MIM)*".

Reaction	**Reactants**		**Products**		**Kinetic constant**
1a	1	2	3		k_{1a}
1b	3		1	2	k_{1b}
2a	3	3	4		k_{2a}
2b	4		3	3	k_{2b}
3	4		5		k_3
4a	5	6	7		k_{4a}
4b	7		5	6	k_{4b}
5	7		8		k_5
6a	5	9	8		k_{6a}
6b	8		5	9	k_{6b}
7a	10	11	12		k_{7a}
7b	12		10	11	k_{7b}
8a	8	12	13		k_{8a}
8b	13		8	12	k_{8b}
9a	13	15	15a		k_{9a}
9b	15a		13	15	k_{9b}
9c	15a		13	16	k_{9c}
10	16		15		k_{10}

Concerning kinetic constants, we will not assign values to them [6]. However, we assume that they respect the original meaning of the pathway. As for species' names, we will not assign kinetic constants to reactions which are not numbered in the original map.

The only membrane of our concern is the Plasma membrane, thus, we assume an existing external membrane indexed 0 and we assign index 1 to the plasma membrane. The resulting, fairly simple, Membrane Structure is given by:

$$\{\langle 0,\{1\}\rangle,\langle 1,\varnothing\rangle\}$$

We assume that initially, only species EGF,$EGFR$,Shc,$Grb2$,SOS,Ras:GDP are present. We encode the membrane structure and the possible initial species in the following Tree Structure:

$$\begin{aligned}TS = \{Child_0 &= \langle 0,\{1\},\varnothing,\{(EGF,3)\}\rangle,\\ Child_1 &= \langle 1,\varnothing,\{(EGFR,3),(Ras{:}GDP,2)\},\\ &\qquad \{(Shc,2),(SOS,2),(Grb2,2)\}\rangle\}\end{aligned}$$

The corresponding initial term is given by:

$$\begin{aligned}&T_0 \equiv [\![TS]\!]_T = [\![Child_0]\!] =\\ &\quad (0)^L \rfloor (EGF \mid EGF \mid EGF \mid [\![Child_1]\!]) =\\ &\quad (0)^L \rfloor (EGF \mid EGF \mid EGF \mid (1 \mid EGFR \mid EGFR \mid EGFR \mid\\ &\quad Ras{:}GDP \mid Ras{:}GDP)^L \rfloor (Shc \mid Shc \mid SOS \mid SOS \mid Grb2 \mid Grb2))\end{aligned}$$

Now, we proceed with the encoding of reactions into the intermediate representation. We will follow the reactions of table 3.1, thus we here model catalysis as three reactions and for each reaction R_j, the corresponding set of contingencies will be of the kind $C_j = \langle\varnothing, k_j\rangle$. We assume that each reaction is 1–by–1, that is the stoichiometric coefficients (the quantities) of reactants and products are equal to 1.

$$\begin{aligned}RS = \{R_{1a} &= \langle C_{1a},\{\langle EGF,(0,1),1\rangle,\langle EGFR,(1,0),1\rangle\},\\ &\qquad \{\langle EGF{:}EGFR,(1,0),1\rangle\},MS\rangle,\\ R_{1b} &= \langle C_{1b},\{\langle EGF{:}EGFR,(1,0),1\rangle\},\\ &\qquad \{\langle EGF,(0,1),1\rangle,\langle EGFR,(1,0),1\rangle\},MS\rangle\\ R_{2a} &= \langle C_{2a},\{\langle EGF{:}EGFR,(1,0),2\rangle\},\\ &\qquad \{\langle 2EGF{:}EGFR,(1,0),1\rangle\},MS_1\rangle\end{aligned}$$

[6] in practice, initial kinetic constant are only experimental values, which will be adjusted to reasonable ones after several simulation sessions

Table 3.2 CLS+ names and initial concentrations (i.c.) assigned to species 1–16 of Figure 3.9

Species ID	CLS+ Name	i.c.	Extended name
1	EGF	3	Epidermal Growth Factor
2	$EGFR$	3	Epidermal Growth Factor Receptor
3	$EGF{:}EGFR$	0	EGF:EGFR complex
4	$2EGF{:}EGFR$	0	EGF:EGFR dimer
5	$2EGF{:}pEGFR$	0	EGF:phosphorylated–EGFR dimer
6	Shc	2	Src homology 2 domain containing transforming protein 1
7	$2EpE{:}Shc$	0	(EGF:phosphorylated–EGFR dimer):Shc
8	$2EpE{:}pShc$	0	(EGF:phosphorylated–EGFR dimer):phosphorylated–Shc
9	$pShc$	0	phosphorylated–Shc
10	$Grb2$	2	Growth factor receptor-bound protein 2
11	SOS	2	A guanine nucleotide exchange factor for Ras that binds to GRB2
12	$Grb2{:}SOS$	0	Grb2:SOS complex
13	$2pEpShc{:}GS$	0	((EGF:phosphorylated–EGFR dimer):phosphorylated–Shc):(Grb2:SOS)
14	Ras	0	Transforming protein p21
15	$Ras{:}GDP$	2	Ras:GDP
15a	$2ESGS{:}RGDP$	0	(((EGF:phosphorylated–EGFR dimer):phosphorylated–Shc):(Grb2:SOS)):(Ras:GDP)
16	$Ras{:}GTP$	0	Ras:GTP

$$
\begin{aligned}
R_{2b} &= \langle C_{2b}, \{\langle 2EGF{:}EGFR, (1,0), 1\rangle\},\\
&\qquad \{\langle EGF{:}EGFR, (1,0), 2\rangle\}, MS_1\rangle\\
R_3 &= \langle C_3, \{\langle 2EGF{:}EGFR, (1,0), 1\rangle\},\\
&\qquad \{\langle 2EGF{:}pEGFR, (1,0), 1\rangle\}, MS_1\rangle\\
R_{4a} &= \langle C_{4a}, \{\langle 2EGF{:}pEGFR, (1,0), 1\rangle, \langle Shc, (1,1), 1\rangle\},\\
&\qquad \{\langle 2EpE{:}Shc, (1,0), 1\rangle\}, MS_1\rangle\\
R_{4b} &= \langle C_{4b}, \{\langle 2EpE{:}Shc, (1,0), 1\rangle\},\\
&\qquad \{\langle 2EGF{:}EGFR, (1,0), 1\rangle, \langle Shc, (1,1), 1\rangle\}, MS_1\rangle\\
R_5 &= \langle C_5, \{\langle 2EpE{:}Shc, (1,0), 1\rangle\},\\
&\qquad \{\langle 2EpE{:}pShc, (1,0), 1\rangle\}, MS_1\rangle\\
R_{6a} &= \langle C_{6a}, \{\langle 2EGF{:}pEGFR, (1,0), 1\rangle \langle pShc, (1,1), 1\rangle\},\\
&\qquad \{\langle 2EpE{:}pShc, (1,0), 1\rangle\}, MS_1\rangle\\
R_{6b} &= \langle C_{6b}, \{\langle 2EpE{:}pShc, (1,0), 1\rangle\},\\
&\qquad \{\langle 2EGF{:}pEGFR, (1,0), 1\rangle \langle pShc, (1,1), 1\rangle\}, MS_1\rangle\\
R_{7a} &= \langle C_{7a}, \{\langle Grb2, (1,1), 1\rangle, \langle SOS, (1,1), 1\rangle\},\\
&\qquad \{\langle Grb2{:}SOS, (1,1), 1\rangle\}, MS_1\rangle,\\
R_{7b} &= \langle C_{7b}, \{\langle Grb2{:}SOS, (1,1), 1\rangle\},\\
&\qquad \{\langle Grb2, (1,1), 1\rangle, \langle SOS, (1,1), 1\rangle\}, MS_1\rangle\\
R_{8a} &= \langle C_{8a}, \{\langle 2EpE{:}pShc, (1,0), 1\rangle, \langle Grb2{:}SOS, (1,1), 1\rangle\},
\end{aligned}
$$

$$
\begin{aligned}
&\{\langle 2pEpShc{:}GS, (1,0), 1\rangle\}, MS_1\rangle \\
R_{8b} &= \langle C_{8b}, \{\langle 2pEpShc{:}GS, (1,0), 1\rangle\}, \\
&\{\langle 2EpE{:}pShc, (1,0), 1\rangle, \langle Grb2{:}SOS, (1,1), 1\rangle\}, MS_1\rangle \\
R_{9a} &= \langle C_{9a}, \{\langle 2pEpShc{:}GS, (1,0), 1\rangle, \langle Ras{:}GDP, (1,0), 1\rangle\}, \\
&\{\langle 2ESGS{:}RGDP, (1,0), 1\rangle\}, MS_1\rangle \\
R_{9b} &= \langle C_{9b}, \{\langle 2ESGS{:}RGDP, (1,0), 1\rangle\}, \\
&\{\langle 2pEpShc{:}GS, (1,0), 1\rangle, \langle Ras{:}GDP, (1,0), 1\rangle\}, MS_1\rangle \\
R_{9c} &= \langle C_{9c}, \{\langle 2ESGS{:}RGDP, (1,0), 1\rangle\}, \\
&\{\langle 2pEpShc{:}GS, (1,0), 1\rangle, \langle Ras{:}GTP, (1,0), 1\rangle\}, MS_1\rangle \\
R_{10} &= \langle C_{10}, \{\langle Ras{:}GTP, (1,0), 1\rangle\}, \\
&\{\langle Ras{:}GDP, (1,0), 1\rangle\}, MS_1\rangle\}
\end{aligned}
$$

where $MS = \{\langle 0, \{1\}\rangle, \langle 1, \varnothing\rangle\}$, $MS_1 = \{\langle 1, \varnothing\rangle\}$

The corresponding Set of Stochastic CLS+ Rewrite Rules, RR is given by applying the translation function to the Set of Reactions RS:

$$[\![RS]\!]_{rr} = \{[\![R_{1a}]\!]_r, [\![R_{1b}]\!]_r, \ldots, [\![R_{10}]\!]_r\}$$

In this example, all rate functions are indeed constants, thus the triple of a rewrite rule contains directly the rate constant. We list the obtained rewrite rules.

$$
\begin{aligned}
[\![R_{1a}]\!]_r = (P1_{1a} &= \big(0 \,|\, X_0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, EGFR \,|\, X_2\big)^L \rfloor X_3 \,|\, X_1), \\
P2_{1a} &= \big(0 \,|\, X_0\big)^L \rfloor ((1 \,|\, EGF{:}EGFR \,|\, X_2\big)^L \rfloor X_3 \,|\, X_1), \\
k_{1a}) &
\end{aligned}
$$

$$
\begin{aligned}
[\![R_{1b}]\!]_r = (P1_{1b} &= \big(0 \,|\, X_0\big)^L \rfloor ((1 \,|\, EGF{:}EGFR \,|\, X_2\big)^L \rfloor X_3 \,|\, X_1), \\
P2_{1b} &= \big(0 \,|\, X_0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, EGFR \,|\, X_2\big)^L \rfloor X_3 \,|\, X_1), \\
k_{1b}) &
\end{aligned}
$$

$$
\begin{aligned}
[\![R_{2a}]\!]_r = (P1_{2a} &= \big(1 \,|\, EGF{:}EGFR \,|\, EGF{:}EGFR \,|\, X_0\big)^L \rfloor X_1, \\
P2_{2a} &= \big(1 \,|\, 2EGF{:}EGFR \,|\, X_0\big)^L \rfloor X_1, \\
k_{2a}) &
\end{aligned}
$$

$$
\begin{aligned}
[\![R_{2b}]\!]_r = (P1_{2b} &= \big(1 \,|\, 2EGF{:}EGFR \,|\, X_0\big)^L \rfloor X_1, \\
P2_{2b} &= \big(1 \,|\, EGF{:}EGFR \,|\, EGF{:}EGFR \,|\, X_2\big)^L \rfloor X_1, \\
k_{2b}) &
\end{aligned}
$$

$$
\begin{aligned}
[\![R_3]\!]_r = (P1_3 &= \big(1 \,|\, 2EGF{:}EGFR \,|\, X_0\big)^L \rfloor X_1, \\
P2_3 &= \big(1 \,|\, 2EGF{:}pEGFR \,|\, X_0\big)^L \rfloor X_1, \\
k_3) &
\end{aligned}
$$

$$
\begin{aligned}
[\![R_{4a}]\!]_r = (P1_{4a} &= \big(1 \,|\, 2EGF{:}pEGFR \,|\, X_0\big)^L \rfloor (Shc \,|\, X_1), \\
P2_{4a} &= \big(1 \,|\, 2EpE{:}Shc \,|\, X_0\big)^L \rfloor X_1,
\end{aligned}
$$

$$
\begin{aligned}
&k_{4a})\\
[\![R_{4b}]\!]_r = (&P1_{4b} = \big(1 \,|\, 2EpE{:}Shc \,|\, X_0\big)^L \rfloor X_1,\\
&P2_{4b} = \big(1 \,|\, 2EGF{:}pEGFR \,|\, X_0\big)^L \rfloor (Shc \,|\, X_1),\\
&k_{4b})\\
[\![R_{5}]\!]_r = (&P1_{5} = \big(1 \,|\, 2EpE{:}Shc \,|\, X_0\big)^L \rfloor X_1,\\
&P2_{5} = \big(1 \,|\, 2EpE{:}pShc \,|\, X_0\big)^L \rfloor X_1,\\
&k_{5})\\
[\![R_{6a}]\!]_r = (&P1_{6a} = \big(1 \,|\, 2EGF{:}pEGFR \,|\, X_0\big)^L \rfloor (pShc \,|\, X_1),\\
&P2_{6a} = \big(1 \,|\, 2EpE{:}pShc \,|\, X_0\big)^L \rfloor X_1,\\
&k_{6a})\\
[\![R_{6b}]\!]_r = (&P1_{6b} = \big(1 \,|\, 2EpE{:}pShc \,|\, X_0\big)^L \rfloor X_1,\\
&P2_{6b} = \big(1 \,|\, 2EGF{:}pEGFR \,|\, X_0\big)^L \rfloor (pShc \,|\, X_1),\\
&k_{6b})\\
[\![R_{7a}]\!]_r = (&P1_{7a} = \big(1 \,|\, X_0\big)^L \rfloor (Grb2 \,|\, SOS \,|\, X_1),\\
&P2_{7a} = \big(1 \,|\, X_0\big)^L \rfloor (Grb2{:}SOS \,|\, X_1),\\
&k_{7a})\\
[\![R_{7b}]\!]_r = (&P1_{7b} = \big(1 \,|\, X_0\big)^L \rfloor (Grb2{:}SOS \,|\, X_1),\\
&P2_{7b} = \big(1 \,|\, X_0\big)^L \rfloor (Grb2 \,|\, SOS \,|\, X_1),\\
&k_{7b})\\
[\![R_{8a}]\!]_r = (&P1_{8a} = \big(1 \,|\, 2EpE{:}pShc \,|\, X_0\big)^L \rfloor (Grb2{:}SOS \,|\, X_1),\\
&P2_{8a} = \big(1 \,|\, 2pEpShc{:}GS \,|\, X_0\big)^L \rfloor X_1,\\
&k_{8a})\\
[\![R_{8b}]\!]_r = (&P1_{8b} = \big(1 \,|\, 2pEpShc{:}GS \,|\, X_0\big)^L \rfloor X_1,\\
&P2_{8b} = \big(1 \,|\, 2EpE{:}pShc \,|\, X_0\big)^L \rfloor (Grb2{:}SOS \,|\, X_1),\\
&k_{8b})\\
[\![R_{9a}]\!]_r = (&P1_{9a} = \big(1 \,|\, 2pEpShc{:}GS \,|\, Ras{:}GDP \,|\, X_0\big)^L \rfloor X_1,\\
&P2_{9a} = \big(1 \,|\, 2ESGS{:}RGDP \,|\, X_0\big)^L \rfloor X_1,\\
&k_{9a})\\
[\![R_{9b}]\!]_r = (&P1_{9b} = \big(1 \,|\, 2ESGS{:}RGDP \,|\, X_0\big)^L \rfloor X_1,\\
&P2_{9b} = \big(1 \,|\, 2pEpShc{:}GS \,|\, Ras{:}GDP \,|\, X_0\big)^L \rfloor X_1,\\
&k_{9b})\\
[\![R_{9c}]\!]_r = (&P1_{9c} = \big(1 \,|\, 2ESGS{:}RGDP \,|\, X_0\big)^L \rfloor X_1,\\
&P2_{9c} = \big(1 \,|\, 2pEpShc{:}GS \,|\, Ras{:}GTP \,|\, X_0\big)^L \rfloor X_1,
\end{aligned}
$$

$$
\begin{aligned}
& k_{9c}) \\
[\![R_{10}]\!]_r = (& P1_{10} = \big(1 \,|\, \mathit{Ras{:}GTP} \,|\, X_0\big)^L \rfloor X_1, \\
& P2_{10} = \big(1 \,|\, \mathit{Ras{:}GDP} \,|\, X_0\big)^L \rfloor X_1, \\
& k_{10})
\end{aligned}
$$

Now we show a transition path for a possible evolution of this signalling pathway, starting from the initial term T_0, previously determined.

$$
\begin{aligned}
T_0 \equiv & \big(0\big)^L \rfloor (EGF \,|\, EGF \,|\, EGF \,|\, \big(1 \,|\, EGFR \,|\, EGFR \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, Shc \,|\, SOS \,|\, SOS \,|\, Grb2 \,|\, Grb2)) \\
\xrightarrow{(R_{1a})} & \big(0\big)^L \rfloor (EGF \,|\, EGF \,|\, \big(1 \,|\, \mathit{EGF{:}EGFR} \,|\, EGFR \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, Shc \,|\, SOS \,|\, SOS \,|\, Grb2 \,|\, Grb2)) \\
\xrightarrow{(R_{1a})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, \mathit{EGF{:}EGFR} \,|\, \mathit{EGF{:}EGFR} \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, Shc \,|\, SOS \,|\, SOS \,|\, Grb2 \,|\, Grb2)) \\
\xrightarrow{(R_{2a})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, 2\mathit{EGF{:}EGFR} \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, Shc \,|\, SOS \,|\, SOS \,|\, Grb2 \,|\, Grb2)) \\
\xrightarrow{(R_{3})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, 2\mathit{EGF{:}pEGFR} \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, Shc \,|\, SOS \,|\, SOS \,|\, Grb2 \,|\, Grb2)) \\
\xrightarrow{(R_{4a})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, 2\mathit{EpE{:}Shc} \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, SOS \,|\, SOS \,|\, Grb2 \,|\, Grb2)) \\
\xrightarrow{(R_{5})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, 2\mathit{EpE{:}pShc} \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, SOS \,|\, SOS \,|\, Grb2 \,|\, Grb2)) \\
\xrightarrow{(R_{7a})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, 2\mathit{EpE{:}pShc} \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, SOS \,|\, \mathit{Grb2{:}SOS} \,|\, Grb2)) \\
\xrightarrow{(R_{8a})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, 2\mathit{pEpShc{:}GS} \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, SOS \,|\, Grb2)) \\
\xrightarrow{(R_{9a})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, 2\mathit{ESGS{:}RGDP} \,|\, EGFR \,| \\
& \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, SOS \,|\, Grb2)) \\
\xrightarrow{(R_{9c})} & \big(0\big)^L \rfloor (EGF \,|\, \big(1 \,|\, 2\mathit{pEpShc{:}GS} \,|\, EGFR \,| \\
& \mathit{Ras{:}GTP} \,|\, \mathit{Ras{:}GDP}\big)^L \rfloor (Shc \,|\, SOS \,|\, \mathit{Grb2{:}SOS} \,|\, Grb2))
\end{aligned}
$$

Looking at this transition path, we can appreciate how $\mathit{Ras{:}GTP}$ is introduced into the cell. Note that we did not make use of rule R_{6a} ad R_{6b}: rule R_{6b} allows a $2\mathit{EpE{:}pShc}$ compound to release the phosphorylated–Shc into the cytosol (the fluid inside the plasma membrane); therefore, in alternative to the shown path, where the compound $2EpE{:}pShc$ is created after transition $\xrightarrow{(R_5)}$, a compound $2EpE{:}pShc$ could also be generated by the binding of a $2EGF{:}pEGFR$ dimer with such a freed phosphorylated–Shc (imagine a state in which there are already several dimers of the kind $2EGF{:}pEGFR$).

The reaction rate of each transition depends on the respective kinetic constants and the concentrations of the reactants. We know from the map annotations that reaction R_{10}, namely the conversion of $Ras{:}GDP$ into $Ras{:}GTP$ is quite slow: this fact should be modeled by assigning a quite small kinetic constant to this rule, while the rules for the enzymatic creation of $Ras{:}GTP$, namely R_{9a}, R_{9c}, should have greater kinetic constants than R_{10}; thus, a transition by R_{10} will be possible, when the cell has accumulated too may $Ras{:}GTP$ molecules and hence it needs to convert some of them back into $Ras{:}GDP$.

Now, we would like to show how, given our semantics for explicit MIMs, it is possible to represent the enzymatic catalysis numbered as reaction 9, in map 3.9, through the specification of a contingency, instead of the three reactions R_{9a}, R_{9b}, R_{9c}. We describe the stoichiometric conversion of $Ras{:}GDP$ into $Ras{:}GTP$ as a reaction having species $2pEpShc{:}GS$ as catalyzer: in the absence of its catalyzer, the reaction is very slow (modeled by k_s); while, in the presence of its catalyzer, the reaction is faster (modeled by k_f). The new contingency set for this reaction is:

$$C_9 = \{\langle k_s, \varnothing\rangle, \langle k_f, \{\langle 2pEpShc{:}GS, (1,0), 1\rangle\}\rangle\}$$

We replace the three reactions R_{9a}, R_{9b}, R_{9c} with a single reaction, named R_9, modeling the just described event.

$$\begin{aligned} R_9 = \langle C_9, &\{\langle Ras{:}GDP, (1,0), 1\rangle\},\\ &\{\langle Ras{:}GTP, (1,0), 1\rangle\}, MS_1\rangle \end{aligned}$$

The corresponding rewrite rule is:

$$\begin{aligned} [\![R_9]\!]_r = (&P1_9 = \big(1 \,|\, Ras{:}GDP \,|\, X_0\big)^L \rfloor X_1,\\ &P2_9 = \big(1 \,|\, Ras{:}GTP \,|\, X_0\big)^L \rfloor X_1,\\ &f) \end{aligned}$$

$$f = \begin{cases} k_f & \text{if } \sigma(X_0) \equiv 2pEpShc{:}GS \,|\, T \\ k_s & \text{otherwise} \end{cases}$$

Since we specified a very low value for k_s, a transition by this rewrite rule does presumably not happen before the formation of complex $2pEpShc{:}GS$.

Take the example of transition path we described before, suppose we reached the following configuration:

$$\begin{aligned} &\vdots \quad \vdots \\ &\xrightarrow{(R_{8a})} \quad (0)^L \rfloor (EGF \mid (1 \mid 2pEpShc{:}GS \mid EGFR \mid \\ &\qquad\quad Ras{:}GDP \mid Ras{:}GDP)^L \rfloor (Shc \mid SOS \mid Grb2)) \end{aligned}$$

The presence of $2pEpShc{:}GS$ let the reaction rate R_9 increase to k_f: its chance of being stochastically executed notably increases. In fact, we have:

$$\sigma(X_0) \equiv 2pEpShc{:}GS \mid EGFR \mid Ras{:}GDP$$

In case the stochastic system chooses R_9 as the reaction to be executed, the system will perform a transition to the following configuration:

$$\begin{aligned} &\xrightarrow{(R_9)} \quad (0)^L \rfloor (EGF \mid (1 \mid 2pEpShc{:}GS \mid EGFR \mid \\ &\qquad\quad Ras{:}GTP \mid Ras{:}GDP)^L \rfloor (Shc \mid SOS \mid Grb2)) \end{aligned}$$

Thus, our formalization of this contingency complies with the semantic of the given model.

3.3 Combinatorial interpretation of a MIM

In the combinatorial interpretation, due to the eventual formation of possibly infinite chain of complexes, it is impossible to list all the possible complexes that will arise in the model from the beginning, therefore, we tried to find a formalization flexible enough to represents a possible infinite sequence of complexations.

In the following, we report the initial progresses of this idea: in particular how the initial term would have been constructed and the idea lying behind the very important complexation reaction and its relative decomplexation. We did not develop a comprehensive formalization for constructing the set of rewrite rules for this interpretation of a map. We rather show how rewrite rules could be easily derived for a "flat" MIM, that is without membranes, and the delicacy of managing membranes especially in the presence of translocation.

3.3.1 Initial Term

We will report our ideas in constructing an initial term representing the elementary species initially appearing in the modeled system and all its membrane hierarchy.

Given a map in its combinatorial interpretation, we need a way to define the initial "LCLS+" (an hypothetical *Plus* extension of LCLS) term representing it. To this purpose, we require a formal description for all molecules and complexes appearing at a given moment. The first thing we do is providing a formal intermediate encoding for the molecular species and the membrane structure appearing in a map.

It is important to note that each species in a map has a set of well defined binding sites. We are interested in giving an unambiguous name to each binding site of each species. A binding between two species will be hence univocally represented by the pair of its binding sites. Remind that in LCLS a binding is represented by a link, therefore, a binding between two species will correspond to the assignment of the same label to a pair of binding sites. Given a map, we assign to each species (both elementary and complex) an univocal name. For each species, we assign an univocal name to each of its intra–domain and interacting sites, which are pointed by a binding arrow (both covalent or not). We also assign each membrane an unambiguous identification number. For instance in map of Figure 3.10 we have given a name to each of such components: the auxiliary external membrane is named 0, the plasma membrane is named 1 and the nucleus membrane is named 2.

The species name is written using the same syntax which defines sequences in LCLS, that is:

$$S \ ::= \ \epsilon \ \Big| \ a \ \Big| \ a^n \ \Big| \ S \cdot S$$

Recall that the set of all species is indicated with $\mathcal{S}$.

A species will be indeed represented by an LCLS sequence, containing the species name followed by both its interacting sites and intra–domains. For instance, species **A** of Figure 3.10 will be represented by the sequence $A \cdot de \cdot di$; Note that **A:B** can form an homodimer, thus it has two binding sites and will be represented by the sequence $A : B \cdot ab1 \cdot ab2$. Note that this choice allows the formation of a chain of **A:B**s. Note that interacting points other than binding sites are not taken into account in the construction of the sequence representing the species.

We retain the same definitions, given in the semantics of explicit maps, of *Multiset of Species* 3.1, *Multiset of Species Translation Function* 3.3, *Tree Structure* 3.2 and *Tree Structure Translation Function* 3.4. As before, we assume the whole MIM to be contained in a dummy membrane of index 0. As before, we assume that the initial term representing the initial state of the system modeled by a MIM contains only elementary species, thus the quantity functions of the multisets of species, will return 0 in case of complex species. Complex species will eventually appear as consequence of the application of some rewrite rule.

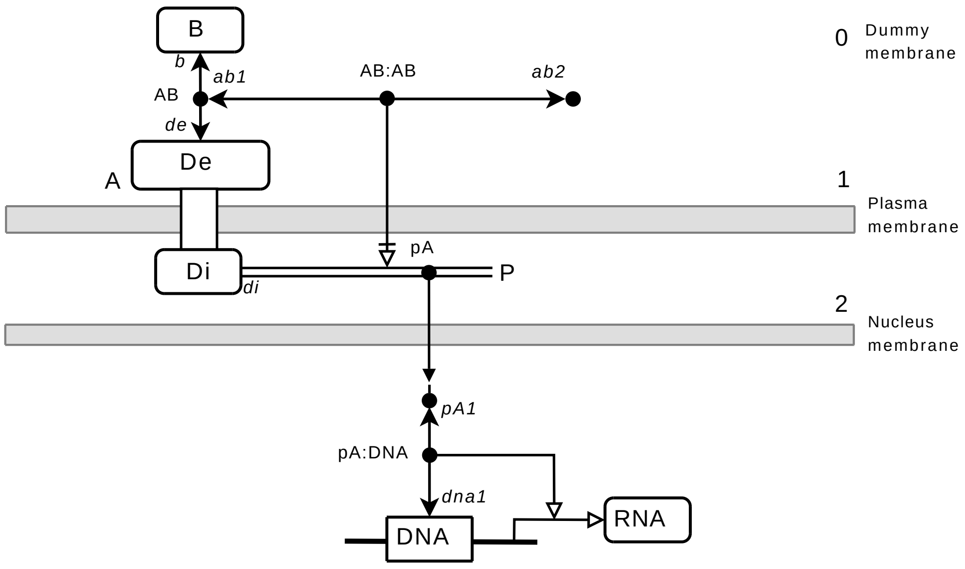

Figure 3.10 Example of named MIM diagram in its combinatorial interpretation: to each species and binding site is assigned an univocal name

We are able to construct an initial LCLS+ term starting from any Molecular Interaction Map. For instance, suppose we want to translate the tree structure of the example of Figure 3.10, which we recall here, where we have assigned a name to each tuple in the set, for simplicity sake:

$$\begin{aligned} TS = \{ & C_0 = \langle 0, \{1\}, \varnothing, (\{B \cdot b\}, m_0)\rangle, \\ & C_1 = \langle 1, \{2\}, (\{A \cdot de \cdot di, AB \cdot ab_1 \cdot ab_2, AB : AB\}, m_1), (\{pA\}, m_2)\rangle, \\ & C_2 = \langle 2, \varnothing, \varnothing, (\{DNA, RNA, pA : DNA\}, m_3)\rangle\} \end{aligned}$$

Assuming that initially only elementary species are presents, we specify the following quantities for each species:

$$\begin{aligned} m_0(B \cdot b) &= 2 \\ m_1(A \cdot de \cdot di) &= 3 \\ m_1(AB \cdot ab1 \cdot ab2) &= 0 \\ m_1(AB : AB) &= 0 \\ m_2(pA \cdot pA1) &= 0 \\ m_3(DNA) &= 1 \\ m_3(RNA) &= 0 \\ m_3(pA : DNA) &= 0 \end{aligned}$$

Then, we will have:

$$[\![TS]\!]_T = [\![C_0]\!] = \big(0\big)^L \rfloor (B \cdot b \,|\, B \cdot b \,|\, [\![C_1]\!]) =$$

$$\big(0\big)^L \rfloor (B \cdot b \,|\, B \cdot b \,|\, \big(1 \,|\, A \cdot de \cdot di \,|\, A \cdot de \cdot di \,|\, A \cdot de \cdot di\big)^L \rfloor ([\![C_2]\!])) =$$
$$\big(0\big)^L \rfloor (B \cdot b \,|\, B \cdot b \,|\, \big(1 \,|\, A \cdot de \cdot di \,|\, A \cdot de \cdot di \,|\, A \cdot de \cdot di\big)^L \rfloor (\big(2\big)^L \rfloor (DNA)))$$

3.3.2 Rewrite Rules

We defined a method to construct the initial term of the model that represents a combinatorial MIM. In this section we make some consideration on how to extrapolate a set of rewrite rules from a given map. As we said before, we do not intend to provide a complete formalization for the set of rewrite rules corresponding to each interaction symbol in a map.

We give the idea lying behind the formalization of a particularly interesting and important class of reactions: binding reactions, which include covalent and non–covalent binding reaction symbols. A binding reaction gives rise to the formation of a complex. We have already seen in section 1.2.4 how the combinatorial interpretation of a MIM could explode in the number of possible complexes, after the addition of a new possible bindings in a map. We need a formal representation for complexes, flexible enough to represent the possible infinite chain of complexations.

The basic idea for bindings, is to state that two species could bind together only through their domains or interacting sites and each domain or sites could interact at most once. As we have seen, each domain and site is given an univocal name and a species is represented by the sequence of its name followed by its domains or sites. A binding is then be univocally represented by an LCLS link between two of these domains of the reactants. If the binding of two reactants results in a new complex, that can eventually bind to another species, through a proper domain, then beside the linking of the reactants' domains, a new sequence representing this complex is created. This new complex is linked to one of the reactants through an auxiliary domain. If the complex resulting from the binding of the two reactants, does not further interact, then we do not create any new sequence for this complex. An appropriate intermediate representation for reactions must keep into account the interacting domains and the new complex species, beside contingencies.

As we have seen, membranes creates compartments. As we did for the explicit interpretation, we reasonably assume that a given reaction can happen only in the place where its symbol appears. Thus, reactions will be strictly connected to a certain membranes, more specifically to a certain membrane surfaces or inner spaces. Concerning combinatorial maps, it could also be interesting to interpret interactions "globally": that is, allowing a reaction to happen in every place, as regards membranes. In this case, we would one reaction rule for each possible binding scenarios (See Figure A.1 of Appendix A for such possible scenarios).

This formalization of bindings through links is fairly simple to represent formation of complexes in the absence of membranes: in this case we have a single compartments and it is any worth to move a species from a place to another.

The formalization of the rewrite rules for binding reactions, in this *flat* combinatorial interpretation, is quite similar to what we did in the explicit interpretation of a MIM: the main differences are that two domains are enough to identify a binding and that we do not have to deal with the spatial position of the reactants, due to the absence of membranes.

In the absence of membranes, the term representing a state of the modeled system, is merely a parallel composition of LCLS sequences. A binding reaction between domains a and b belonging respectively to species $A \cdot a$ and $B \cdot b \cdot bi$, leading to a new complex with a new domain (which we represent with $A{:}B \cdot ab$), is represented by stochastic rewrite rules denoted by:

$$1a. \quad X \,|\, \widetilde{x}_1 \cdot a \cdot \widetilde{y}_1 \,|\, \widetilde{x}_2 \cdot b \cdot \widetilde{y}_2 \overset{f_1}{\mapsto} X \,|\, \widetilde{x}_1 \cdot a^1 \cdot \widetilde{y}_1 \cdot xab^2 \,|\, \widetilde{x}_2 \cdot b^1 \cdot \widetilde{y}_2 \,|\, A{:}B \cdot ab \cdot xab^2$$

$$1b. \quad X \,|\, \widetilde{x}_1 \cdot a^1 \cdot \widetilde{y}_1 \cdot xab^2 \,|\, \widetilde{x}_2 \cdot b^1 \cdot \widetilde{y}_2 \,|\, A{:}B \cdot ab \cdot xab^2 \overset{f_2}{\mapsto} X \,|\, \widetilde{x}_1 \cdot a \cdot \widetilde{y}_1 \,|\, \widetilde{x}_2 \cdot b \cdot \widetilde{y}_2$$

where X models the eventual presence of contingency species affecting the reaction rate of the given reaction, f_1, f_2 are the rate functions and xab is the auxiliary domain which is used to attach the new complex to one of its reactants (and indirectly to all its reactants, following the link notation). We remark that combinatorial maps have some delicate situations that must be clearly interpreted before starting a formalization. For instance, think about interpreting combinatorially the map of Figure 2.2 of section 2.2.2, where **C** can bind to complex **A:B**. We know that the non–covalent binding between **A** and **B** corresponds to two reactions: complexation and decomplexation. Should we consider it possible that complex **A:B** breaks into **A** and **B**, while it is still connected to **C**? It seems reasonable that **A:B** can not break until **C** is connected to it. Our encoding of bindings respects this assumption, in fact: we could encode the non–covalent binding between **A** and **B** with the two rules shown above; supposing that **C** is encoded into $C \cdot c$, the non–covalent binding between **A:B** and **C** through domains ab and c could be encoded by rules denoted by:

$$2a. \quad X \,|\, \widetilde{x}_1 \cdot ab \cdot \widetilde{y}_1 \,|\, \widetilde{x}_2 \cdot c \cdot \widetilde{y}_2 \overset{f_3}{\mapsto} X \,|\, \widetilde{x}_1 \cdot ab^1 \cdot \widetilde{y}_1 \,|\, \widetilde{x}_2 \cdot c^1 \cdot \widetilde{y}_2$$

$$2b. \quad X \,|\, \widetilde{x}_1 \cdot ab^1 \cdot \widetilde{y}_1 \,|\, \widetilde{x}_2 \cdot c^1 \cdot \widetilde{y}_2 X \,|\, \widetilde{x}_1 \cdot ab \cdot \widetilde{y}_1 \,|\, \widetilde{x}_2 \cdot c \cdot \widetilde{y}_2 \overset{f_4}{\mapsto}$$

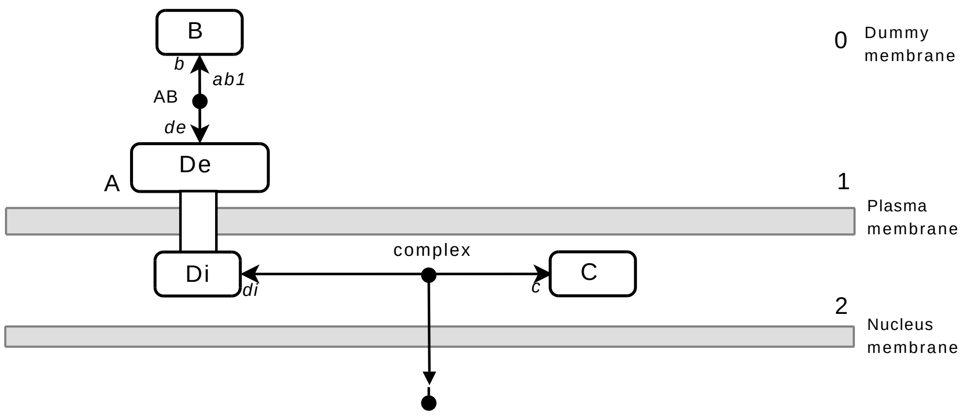

Figure 3.11 Example of translocation in a combinatorially interpreted map: **B** can bind to **A** through the external domain, **C** can bind to **A** through the internal domain; the whole **complex**, which combinatorially could be **A** bound to **C** or **A** bound to both **C** and **B**, could then move to the nucleus.

A state in which complex **A:B:C** exist, could be represented by the following term:

$$T \mid C \cdot c^3 \mid A{:}B \cdot ab^3 \cdot xab^2 \mid A \cdot a^1 \cdot xab^2 \mid B \cdot b^1 \cdot bi$$

We can note that this term can not be matched with rule 1b for decomplexation of $A{:}B$, since domain ab is linked. Thus, $A : B$ can not break down if it is linked to C.

LCLS is expressive enough for representing a possibly infinite chain of complexations in a flat MIM, that is without membranes.

We encounter some hurdle in modeling all events regarding complexes, in the presence of membranes. Keeping in mind that a species could be placed onto a membrane or inside a membrane, one could imagine that the parts forming a complex (its reactants joined with a link) could have been fragmented into several different places belonging to the same compartment or even to different compartments: like one piece onto a membrane, one other piece onto an inner membrane, some other pieces inside the first membrane and so on.

The following example gives a taste of the delicacy of this situation. Please refer to the map of Figure 3.11. Species **A** can bind to **B** through its external domain. **A** can also bind to **C**, which located inside the plasma membrane, through its internal domain. The whole complex can translocate into the nucleus.

The following LCLS term represents **A** in a state where both bonds exist.

$$\left(0\right)^L \rfloor \left(B \cdot b^1 \mid \left(1 \mid A \cdot de^1 \cdot di^2\right)^L \rfloor \left(C \cdot c^2 \mid \left(2\right)^L \rfloor \epsilon\right)\right)$$

How do we translocate such a fragmented complex? We know that, if the inner domain *di* is linked, then A can move to the nucleus, independently from the status of its external domain. Thus, all species connected directly or indirectly to A should be moved to the nucleus. A possible solution would be to scan the LCLS sequence representing the instance of A that is entitled of moving and travel the term representing the whole environment to recollect the terms which are somehow connected to A, that is gathering all the sequences directly or indirectly linked to A. Then, we should move all these terms to the new compartment. These terms would indeed represent kind of a graph structure. Building a function that travels all around a term, looking for links, is not an obvious task. We think that such a function could be provided by the semantics of LCLS.

In giving a semantics for translocation, we would also need the concept of "context". Through contexts, we can explicitly specify only a sub–term of a given term. Context will be useful in formalizing translocation, since a translocation rule will possibly modify the whole term representing the whole model in a certain state. Contexts have already been used in the semantics of Stochastic CLS [29]. We here report a possible adaptation of the definition of contexts for CLS to the *Plus* extension.

Definition 3.21 (Contexts)**.** *Contexts $\mathcal{C}$ are given by the following grammar:*

$$\begin{aligned} \mathcal{C} &::= \square \quad\big|\quad \mathcal{C} \,|\, T \quad\big|\quad T \,|\, \mathcal{C} \quad\big|\quad (S)^L \rfloor \mathcal{C} \quad\big|\quad (\mathcal{BC})^L \rfloor T \\ \mathcal{BC} &::= \square \quad\big|\quad \mathcal{BC} \,|\, S \quad\big|\quad S \,|\, \mathcal{BC} \end{aligned}$$

where $T \in \mathcal{T}$ and $S \in \mathcal{S}$. Contex $\square$ is called the empty context*.*

$C[T]$ (*context application*) denotes the term obtained by replacing $\square$ with T in C, and $C[C']$ (*context composition*) denotes the context obtained by replacing $\square$ with C' in C.

Suppose we have function, which we will call here $graph(S,T)$ (where $S \in \mathcal{S}$), that takes a sequence and a term and returns the Set of Species containing all the sequences in T, which are directly or indirectly linked to S. Suppose that we also have a function, which we call here $move(T, SS, p)$, that takes a term, a set of species and a position and returns a new term in which the species in SS have been moved to the new position p. Suppose we want to translocate of some species S from position $p = (i_1, j_1)$ to $q = (i_2, j_2)$, whose intermediate encoding could be:

$$RS_T \ni R_T = \langle C, \{\langle S, p, 1\rangle\}, \{\langle S, q, 1\rangle\}, MS\rangle$$

where RS_T is the Set of translocation Reactions. Suppose that T is the term representing the whole model in a certain configuration. Suppose that there is a context C for which $C[S] = T$.

We show a possible guideline for the rule for the definition of the semantics of translocation:

$$\frac{R_T \in RS_T \quad SS = graph(S, C[S]) \quad T' = move(C[S], SS \cup \langle S, q, 1\rangle, p)}{(C[S], T', k)}$$

In section 2.4, we saw how LCLS semantics prevents linking between two elements appearing in different compartments: LCLS defines few determined scenarios for possible bindings. In this way, it is granted that binding reactions can happen only if the reactants can indeed be at contact with each other. Thus, a MIM designer should know that reactions, depicted between reactants that are not actually in the same compartment, are not taken into account by the LCLS semantics: such reactions can still be depicted in a map, but they do not affect the system behavior. Indeed, in our experience, we have not encountered any MIM with such kind of reactions. Thus, these LCLS constraints on bindings generally do not imply any practical assumption on MIMs. On the other side, limiting possible bindings to the reasonable ones, could represent a useful means to check the correctness of the various reactions in a MIM.

We remark how contingencies for combinatorial maps, where we keep the information at domain level, could be interpreted in a different way through the use of domains. We could combine a certain instance of some contingency species with a particular instance of some reactant species, through the use of domains. For instance, suppose that C is a contingency affecting a certain complex X. The presence of one molecule of C is entitled to affect the behavior of only one instance of X. Let's do an example. Suppose that we have a reaction of the kind, $A + B \rightarrow A{:}B$, where A can bind to B through their respective domains $(a1, b)$ and C is an inhibitor of this reaction. C can block only one of the possible reactions of this kind, for example, by linking one of its domain to the specific domain of A used for this binding, namely a. Suppose that we have a configuration represented by the following LCLS term:

$$A \cdot a \,|\, A \cdot a \,|\, B \cdot b \,|\, C \cdot c$$

We can represent the fact that one molecule of C inhibits one molecule of A by linking $a2$ to c, in this case the domain of this molecule A is not available for binding to the other reactant B.

$$A \cdot a^1 \,|\, A \cdot a \,|\, B \cdot b \,|\, C \cdot c^1$$

So, C is not entitled to further inhibit and the first instance of A can not bind to B, but another possible binding can still happen, and hence we can reach the following state:

$$A \cdot a^1 \,|\, A \cdot a^2 \,|\, B \cdot b^2 \,|\, C \cdot c^1$$

In this way, the inhibiting power of a C strictly depends on its concentration.

A similar policy can be held in case a certain contingency is necessary for a reaction to happen: requirement can be seen has specular to inhibition. We could probably also model, in a not so simple similar way, catalysis and stimulation, but, anyhow, as we have noticed, these contingencies could be replaced by some reaction symbol.

Chapter 4

Conclusions

In this thesis, we have developed a possible formal semantics for Molecular Interaction Maps (MIMs), by formalizing the translation of MIMs into a model of a proper Calculi of Looping Sequence, whose semantics is based on a transition system. The formal semantics takes into account both quantitative and qualitative aspects of a map. We studied both explicit and combinatorial interpretation of a map. In both cases, given a map, we studied a method to construct a corresponding initial term, representing the possible initial configuration of the model representing the map, and a set of rewrite rules, describing how this initial term can evolve over the time. We used a Stochastic extension of some CLS, which allowed us modeling, through stochastic rewrite rules (rewrite rules combined with a kinetic constant), quantitative aspects of the maps, like frequencies of events and probability of a certain event to happen, following the mass–action law.

We provided a comprehensive formal semantics for explicit maps, through translation into Stochastic CLS+, concluding with a satisfying application of our approach to part of the well–note EGF Signalling pathway map. In particular we shown how, defining a clear and unambiguous semantics for contingency symbols, they can be used in explicit maps aimed at computer simulation, overtaking some of the difficulties, which have been previously encountered by the MIMs' authors their self, in simulating contingencies with standard mathematical approaches based on Ordinary Differential Equations. We remark how, given a map, it is possible to extrapolate a Stochastic CLS+ model from it, by simply encoding the map into an intermediate representation. This intermediate encoding is quite similar to the one already used by biologists, in specifying explicit models aimed at simulation with ODE. Thus, our formal semantics could be a way of getting biologists closer to stochastic simulation approaches. Our formalization allows the encoding of sub–maps, that is, it is possible to encode a determined subset of reactions appearing in a map. In this way, it is possible to study the behavior of part of the map, independently from the rest. A sub–map is simply defined by the set of *labeled* species and reactions in the map: those which are not labeled are not taken into account by our encoding. This choice makes our formalization close to how explicit MIMs have been used in practice. Therefore, the encoding of existing explicit MIMs generally does not require any modification to the original map.

We provided a less formal approach for combinatorial maps, through translation into Stochastic LCLS+, showing possible solutions to the main difficulties, which arise in formalizing the peculiarities of this kind of interpretation. In particular, we gave some guidance on how to model the formation of complexes, which are represented in a compact way in the combinatorial interpretation. In fact, a reaction symbol, in this kind of maps, implicitly describes a possible infinite set of reactions, and hence a possible infinite set of complexes, due to eventual cycles of reactions in the map. Our possible solution is based on an implicit representation of these reactions, thus avoiding to explicitly list in advance all possible complexes and keeping the LCLS+ model dimension proportional to the map dimension. This solution is based on the information about interactions at *domain level*. We also discussed a possible solution to the formalization for translocation: as we have seen, moving a certain complex from a place to another, in the membrane structure, is not an obvious task. Our possible solution requires the definition of a method to gather all the molecules directly or indirectly linked to the molecule which is supposed to move. We remarked how LCLS semantics prevents the definition of not reasonable linking, and hence of not reasonable binding reactions. We gave an alternative interpretation of contingencies exploiting the domain level representation of molecular interactions.

4.1 Related works

We briefly describe some of the approaches, which tried to devise a task similar to the present work.

We already mentioned how Kohn *et al.* [13, 11] encoded explicit maps into a system of differential equations. Once having restricted an explicit map to a subset of MIM symbols (which in practice excludes the concept of contingency), constructing a set of differential equation is not so difficult and there are well founded methods for simulating a system of differential equations. However, we remark how contingencies are not easy to model with differential equations, since they do not have a clear and unambiguous mathematical representation. On the other hand, depicting maps without contingency symbols, results in a noticeable addition of further reactions symbols, which increases the size of the map and can create over–crowding problems. We remark that explicit maps can, with some restriction, be modeled by differential equations. As for combinatorial maps is not even obvious if a possible infinite chain of complexes could be captured by differential equations.

Other approaches, like agent–based or concurrent languages, describe biological interactions in terms of rules, thereby avoiding the combinatorial explosion besetting differential equations. Among these, the k–calculus ([36]), which is a rule–based process algebra, aims to model protein–protein interactions at domain level and provides operations for complexation and activation (modification). In [37], Danos *et al.* illustrate how, using rule–based stochastic strategies, they have been able to simulate some EGF receptor signalling model, which were classically based on differential equations. β–binders [24], strongly inspired by the π-calculus [38], have been used to model a simple example of MIM [34]. In the π–calculus entities running in parallel can communicate through synchronized actions of input and output on named channels. In addition, β–binders enclose entities in boxes, which can interact through the box virtual surface. These boxes can be split or joined at execution time, thus they are suitable to represent biomolecular complexes. A small subset of the MIM for the mammalian cell cycle was modeled using this formalism. In the present work we focus on giving a general formalization for MIMs, taking into account the way in which a map is interpreted (combinatorial explicit). As far as we know, the above cited approaches dealt with "flat" MIMs, without being concerned about compartments created by membranes.

Another work, which tried to devise a general translation of MIMs, was presented by Fonda S. in his Master Thesis [35]. In his thesis, Fonda encodes combinatorial maps into stochastic Concurrent Constraints Programming (sCCP [39]). Fonda makes some assumptions and simplifications on combinatorial maps. Thus, before encoding a map, it is necessary to pre–process it. The pre–processing is aimed at solving delicate situations (similar to the one we shown in section 3.3.2 about decomplexation) and at restricting the amount of symbols in the notation (for instance, stimulation and catalysis are replaced by reaction symbols). The pre–processing consists of a series of rewriting rules applied to the map. The encoding he provided allows the simulation of maps, without having to explicitly specify all possible reactions in advance, thus the encoding does not suffer the possible explosion of reactions in a combinatorial map and is proportional to the map dimension. A complex is represented by a graph–like structure, in which nodes represents molecules and edges the connection among them, at domain level. Each species has a set of univocally named ports (domains or interacting sites) for interconnections. A possible reaction is hence identified by the name of two ports. The evolution of the system follows Gillespie's

Algorithm. The approach is not so different from our interpretation of combinatorial maps, where we represent a possible reaction with two univocally named molecular domains. However, there are significant differences. In Fonda's work, through a system of coordinates, it is possible to identify each instance of each possible molecular species existing at a given moment, while in our encoding it is not possible to distinguish different instances of the same molecular species: our approach reflects the fact that two molecules of the exactly same kind, in the same state (like two EGFR proteins) are usually indistinguishable in nature. In Fonda's work, contingencies are reduced to the only inhibition and requirement; while we provide a possible formalization for all kind of contingencies. A further difference is that Fonda does not deal with membranes: his work concentrates on the encoding of "flat" MIMs; while, in the present work, we try to deal with locality problems of species and interactions, due to the their spatial positioning into a membrane structure.

4.2 Future works

Stochastic CLS is already equipped with a *simulator*. As shown in [29] CLS+ can be encoded into CLS. Thus, using the here presented formal semantics, it should be possible to simulate the stochastic behavior of a given map by encoding it into a Stochastic CLS model. Results gathered during the simulation could be then graphically visualized, eventually even at real–time.

As we have seen, the intermediate encoding of a map, used by our semantics to build a CLS model, is quite similar to the one used by biologists to simulate explicit maps through differential equations. Thus, biologists can easily get familiar with this intermediate encoding and use it to perform simulations through the corresponding CLS model. The translation step from the intermediate encoding to the CLS model should be performed automatically through some software implementation.

We have cited Kitano's Process Diagrams [4, 5] as a diagrammatic notation for biomolecular interactions similar to Kohn's MIM notation. Process diagrams are indeed quite similar to explicit maps and depicts specific pathways. We believe that, following the ideas introduced in the present work, it should be fairly easy to provide a formal semantics for process diagrams into some Stochastic Calculi of Looping Sequences. Process Diagrams are equipped with a computer-aided design (CAD)-like software to depict such diagrams, namely CellDesigner [7]. Through the graphical interface, provided by this software, it could be possible to automatically construct an intermediate representation of the designed map, then the intermediate representation could be automatically encoded into the corresponding CLS model.

In alternative, a graphical interface could eventually be provided to depict MIMs: once a map has been designed through the graphical interface, the system could generate automatically all the structures for the intermediate representation of a map, like species, membrane structure and interactions.

Hence, a CAD tool can be a further meeting point between biologist and stochastic simulations: biologists could simulate maps through stochastic CLS by simply drawing them into a CAD software, labeling the desired interactions, adding some information about kinetic constants and executing some simulation software.

Some work could also be done to complete the formal semantics given for combinatorial maps, describing in detail a formal method for constructing the set of rewrite rules, including means for translocating species from a compartment to another. In this way we can also perform stochastic simulation of combinatorial maps.

It could also be interesting to deal with a dynamic membrane structure: we have seen how CLS can easily model membrane modifications, like breaking, joining, fusion, etc.; even though MIMs do not provide a well defined notation for depicting such membrane modifications (probably, a membrane modification should be represented by a transition from a MIM to a new one), we consider this aspect worth a future investigation.

Appendix A

LCLS semantics

LCLS is an extension of CLS, which adds the possibility of modeling interactions at domain level through the introduction of the concept of link. Links are strictly connected to the concept of compartments created by looping sequences. LCLS introduces labels on basic symbols as a new construct. A label is written as an index at the right top of a basic symbol. For simplicity's sake, labels consist of natural numbers. The followings are the basic principles of LCLS:

- two symbols in a term, having the same label, represent a domain binding;
- a basic element can have no more than one label, which means that a single domain can interact at most with one other domain;
- in a term, modeling the whole system, there can not be labels appearing only once, that is, links must be complete: the presence of unmatched labels within a term means that the term describes only a portion of the modeled system, and that the term is indeed a subterm of a term representing the whole system, which will contain a matching label for that symbol;
- links must respect compartments created by membranes: two species can be linked together, if and only if they belong to the same compartment, which means that these species "can see each other" and are not separated by any membrane; therefore, elements inside a membrane can be linked either to elements inside the membrane or to elements on the membrane itself and elements inside a looping sequence can not be linked to elements outside.

Figure A.1 shows some example of LCLS possible links, in a schematic representation of molecules and membranes, multi–domain molecules are depicted with small adjacent circles, one for each domain. The correspondent

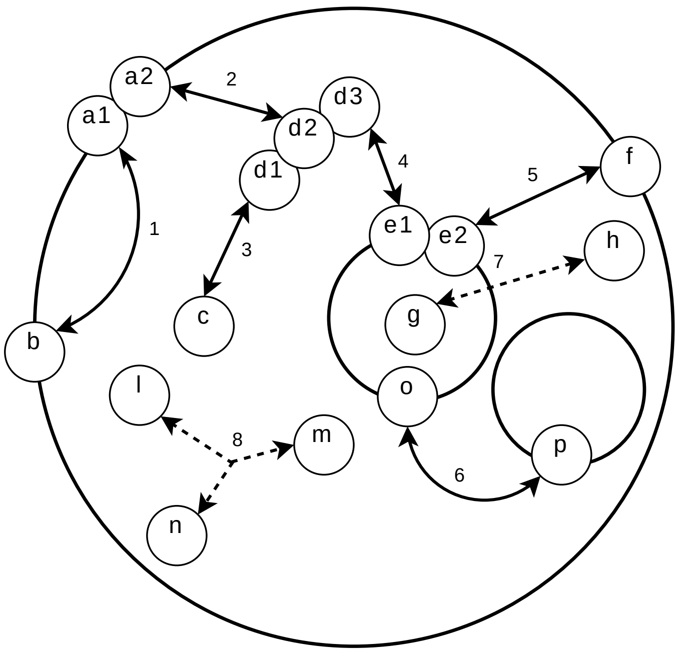

Figure A.1 Schema of possible Links – dashed lines represent prohibited links. The correspondent LCLS term is $\big(a_1{}^1 \cdot a_2{}^2 \cdot f^5 \cdot b^1\big)^L \rfloor (d_1{}^3 \cdot d_2{}^2 \cdot d_3{}^4 \,|\, c^3 \,|\, l^8 \,|\, n^8 \,|\, m^8 \,|\, h^7 \big(e_1{}^4 \cdot e_2{}^5 \cdot o^6\big)^L \rfloor g^7 \,|\, \big(p^6\big)^L \rfloor \epsilon)$

LCLS term, as we shall see, is not "well–formed", since it contains prohibited links. It is worth noticing that there are some recurrent scenarios of possible bindings:

1. binding between elements of the same sequence, e.g. link 1;
2. binding between elements of distinct parallel sequences, e.g. link 3;
3. binding between an element standing on a membrane and an element inside this membrane, e.g. link 2;
4. binding between an element standing on a membrane and an element outside this membrane, e.g. link 4;
5. binding between elements standing on nested membranes, e.g. link 5;
6. binding between elements standing on distinct parallel membranes, e.g. link 6;

These scenarios are considered in the formulation of a semantics for LCLS.

Note that multi-domain molecule $a_1{}^1 \cdot a_2{}^2$ standing on the external membrane, will not be semantically separated from the other molecules standing on the same membrane, indeed they are all concatenated in a sequence. This is due to the fact that LCLS is indeed a CLS extension

In the following we will recall the formal semantics of LCLS

The following is the syntax of terms of LCLS.

Definition A.1 (Terms)**.** Terms T *and* Sequences S *of LCLS are given by the following grammar:*

$$\begin{array}{rcl} T & ::= & S \;\;\big|\;\; \left(S\right)^L \rfloor T \;\;\big|\;\; T \,|\, T \\ S & ::= & \epsilon \;\;\big|\;\; a \;\;\big|\;\; a^n \;\;\big|\;\; S \cdot S \end{array}$$

where a is a generic element of $\mathcal{E}$, and n is a natural number. The infinite set of terms is denoted by $\mathcal{T}$, and the infinite set of sequences is denoted by $\mathcal{S}$.

In order to consider as equivalent two syntactically different terms, the same structural congruence relation defined for CLS is adopted.

Definition A.2 (Structural Congruence)**.** *The structural congruence relations $\equiv_S$ and $\equiv_T$ are the least congruence relations on sequences and on terms, respectively, satisfying the following rules:*

$$S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3 \qquad S \cdot \epsilon \equiv_S \epsilon \cdot S \equiv_S S$$

$$S_1 \equiv_S S_2 \textit{ implies } S_1 \equiv_T S_2 \textit{ and } \left(S_1\right)^L \rfloor T \equiv_T \left(S_2\right)^L \rfloor T$$

$$T_1 \,|\, T_2 \equiv_T T_2 \,|\, T_1 \qquad T_1 \,|\, (T_2 \,|\, T_3) \equiv_T (T_1 \,|\, T_2) \,|\, T_3 \qquad T \,|\, \epsilon \equiv_T T$$

$$\left(\epsilon\right)^L \rfloor \epsilon \equiv \epsilon \qquad \left(S_1 \cdot S_2\right)^L \rfloor T \equiv_T \left(S_2 \cdot S_1\right)^L \rfloor T$$

In addition, two terms which differ only for the name of their links should be considered as equivalent. To this purpose, a renaming function and an equivalence relation are defined, which will be described further on (the so called, α–renaming function and α–equivalent relation, respectively).

In order to define patterns of LCLS, a construct for labeled element variables is introduced in sequence patterns.

Definition A.3 (Patterns)**.** Patterns P *and* sequence patterns SP *of LCLS are given by the following grammar:*

$$\begin{array}{rcl} P & ::= & SP \;\;\big|\;\; \left(SP\right)^L \rfloor P \;\;\big|\;\; P \,|\, P \;\;\big|\;\; X \\ SP & ::= & \epsilon \;\;\big|\;\; a \;\;\big|\;\; a^n \;\;\big|\;\; SP \cdot SP \;\;\big|\;\; \widetilde{x} \;\;\big|\;\; x \;\;\big|\;\; x^n \end{array}$$

where a is a generic element of $\mathcal{E}$, n is a natural number and $X, \widetilde{x}$ and x are generic elements of TV, SV and $\mathcal{X}$, respectively. The infinite set of patterns is denoted by $\mathcal{P}$.

Note that, since terms are also patterns (ground patterns), everything defined for patterns is also valid for terms.

A formal method is needed to check whether a pattern (and hence a term) is *well–formed*, i.e. the pattern respects the basic principles listed previously, on which LCLS relies. To this purpose, the notion of *compartment* and of *top–level compartment* of a pattern are given:

Compartment: given a pattern, each of its subpatterns, which is contained in a looping sequence and whose content is ignored, is a compartment;

Top–level compartment: given a pattern, the portion of it, which is not contained in any looping sequence, is the top–level compartment of the pattern.

Let‘s report an example, taken from [30] in order to clarify these notions. Given

$$P = a \mid \big(b\big)^L \rfloor c \mid \big(d\big)^L \rfloor (X \mid \big(e\big)^L \rfloor f)$$

we have:

- top–level compartment of P : $\quad a \mid \big(b\big)^L \rfloor \epsilon \mid \big(d\big)^L \rfloor \epsilon$
- other compartments of P : $\quad c, \qquad X \mid \big(e\big)^L \rfloor \epsilon, \qquad f$

Informally, “an LCLS pattern is well–formed if and only if a label occurs no more than twice, and two occurrences of a label are always in the same compartment” [30]. A *type system* is used to derive the well–formedness of a pattern. The type system is defined by a set of inference rules, whose conclusion has the form $(N, N') \models P$, where N and N' are two sets of natural numbers.

N is the set of labels appearing twice in the top–level compartment of P;

N' is the set of labels appearing once in the top–level compartment of P.

Definition A.4 (Type System)**.** *The typing algorithm for LCLS patterns is defined by the following inference rules:*

1. $\big(\varnothing,\varnothing\big) \models \epsilon$ 2. $\big(\varnothing,\varnothing\big) \models a$ 3. $\big(\varnothing,\{n\}\big) \models a^n$

4. $\big(\varnothing,\varnothing\big) \models x$ 5. $\big(\varnothing,\{n\}\big) \models x^n$ 6. $\big(\varnothing,\varnothing\big) \models \widetilde{x}$ 7. $\big(\varnothing,\varnothing\big) \models X$

8. $$\frac{\big(N_1,N_1'\big) \models SP_1 \;\; \big(N_2,N_2'\big) \models SP_2 \;\; N_1 \cap N_2 = N_1' \cap N_2 = N_1 \cap N_2' = \varnothing}{\big(N_1 \cup N_2 \cup (N_1' \cap N_2'), (N_1' \cup N_2') \setminus (N_1' \cap N_2')\big) \models SP_1 \cdot SP_2}$$

9. $$\frac{\big(N_1,N_1'\big) \models P_1 \;\; \big(N_2,N_2'\big) \models P_2 \;\; N_1 \cap N_2 = N_1' \cap N_2 = N_1 \cap N_2' = \varnothing}{\big(N_1 \cup N_2 \cup (N_1' \cap N_2'), (N_1' \cup N_2') \setminus (N_1' \cap N_2')\big) \models P_1 \mid P_2}$$

10. $$\frac{\big(N_1,N_1'\big) \models SP \;\; \big(N_2,N_2'\big) \models P \;\; N_1 \cap N_2 = N_1' \cap N_2 = N_1 \cap N_2' = \varnothing \;\; N_2' \subseteq N_1'}{\big(N_1 \cup N_2', N_1' \setminus N_2'\big) \models \big(SP\big)^L \rfloor P}$$

where a is a generic element of $\mathcal{E}$, n is a natural number, and $X, \widetilde{x}$ and x are generic elements of TV, SV and $\mathcal{X}$, respectively and P_1, P_2 are any pattern, SP_1, SP_2 are any sequence pattern. We write $\models P$ if there exist $N, N' \subset \mathbb{N}$ such that $(N, N') \models P$, and $\not\models P$ otherwise.

Rules 1–7 state, quite obviously, the well–typedness of basic sequence patterns: in case of labeled elementary symbol, N‘ is the set containing its label.

Rule 8 states that a sequence pattern $P = SP_1 \cdot SP_2$ is well-typed if there are no labels occurring either four times ($N_1 \cap N_2 = \varnothing$) or three times ($N_1' \cap N_2 = N_1 \cap N_2' = \varnothing$) in P; in case P is well-typed, the set of labels occurring twice in it contains all the labels occurring twice in either SP_1 or in SP_2, plus the labels occurring once in both subpatterns, that is $N_1 \cup N_2 \cup (N_1' \cap N_2')$; while, the set of labels occurring once in P contains the labels which occur once either in SP_1 or in SP_2, excluding those which appear in both subpatterns (note that these are a link in P), that is $(N_1' \cup N_2') \setminus (N_1' \cap N_2')$;

Rule 9 states the same as rule 8, but for the parallel composition;

Rule 10 states that the labels which occur only once in P, must also occur once in SP, in other words, elements inside the membrane SP have their last chance of binding with elements on the membrane itself (remind that elements inside a membrane can not bind to elements outside the membrane itself), that is $N_2' \subseteq N_1'$, otherwise $\big(SP\big)^L \rfloor P$ is not well-typed. Note that $\big(SP\big)^L \rfloor P$ will be typed using labels which are used to type SP and which are not used to type P, therefore, labels forming a link in P will be “forgotten” by the typing algorithm, this allows us to *reuse* such labels in outside the looping sequence

Note that, rule 6 and 7 state that sequence variables and term variables do not contain any label: this assumption can result “strange”, since pattern variables could be eventually be instantiated with terms containing link labels (keep in mind that a pattern represents a set of terms, in particular, all those terms for which an instantiation function σ exists such that $P\sigma$ is a well–formed term). Therefore, we think that well–formedness “makes sense” only on terms, since we do not know yet how a variable will be instantiated. However, we will retain the definition of well–formedness on patterns as found on [30], since this will not affect any consideration on well–formedness of terms.

We report a very useful typing example taken from [30].

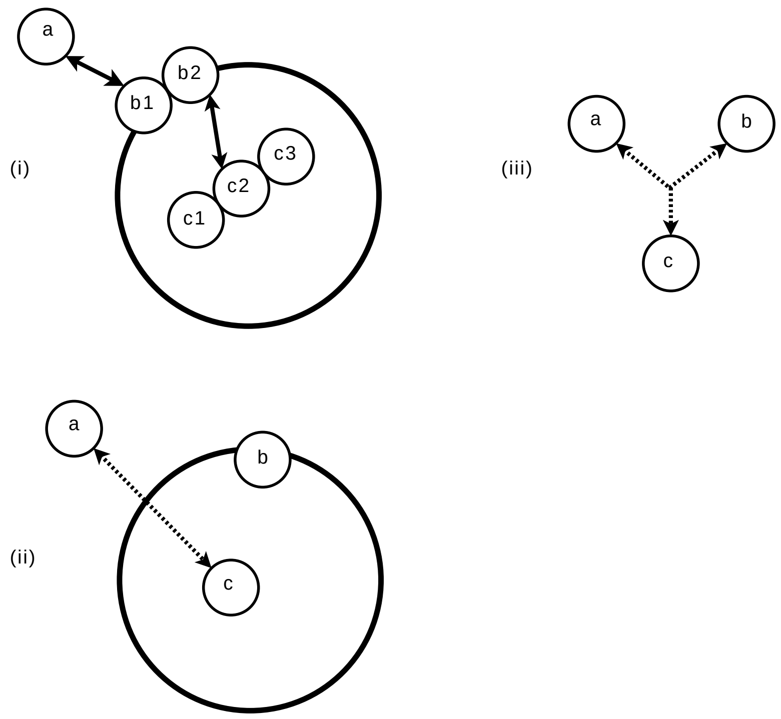

Figure A.2 Example of well–formed and non–well–formed patterns in LCLS: (i) represents $a^1 \mid \big(b1^1 \cdot b2^2\big)^L \rfloor c1 \cdot c2^2 \cdot c3$; (ii) represents $a^1 \mid \big(b\big)^L \rfloor c^1$; (iii) represents $a^1 \mid b^1 \mid c^1$.

The pattern $a^1 \mid \big(b1^1 \cdot b2^2\big)^L \rfloor c1 \cdot c2^2 \cdot c3$ (visually represented in Fig. A.2.(i)) can be typed as follows.

$$
\begin{array}{rll}
(\varnothing, \{1\}) \models a^1 & & \text{by rule 3} \\
(\varnothing, \{1,2\}) \models b1^1 \cdot b2^2 & & \text{by rules 3 and 8} \\
(\varnothing, \{2\}) \models c1 \cdot c2^2 \cdot c3 & & \text{by rules 2, 3 and 8} \\
(\{2\}, \{1\}) \models \big(b1^1 \cdot b2^2\big)^L \rfloor c1 \cdot c2^2 \cdot c3 & & \text{by rule 10} \\
(\{1,2\}, \varnothing) \models a^1 \mid \big(b1^1 \cdot b2^2\big)^L \rfloor c1 \cdot c2^2 \cdot c3 & & \text{by rule 9}
\end{array}
$$

The pattern $a^1 \mid \big(b\big)^L \rfloor c^1$ (represented in Fig. A.2.(ii)) cannot be typed. In fact

$$
\begin{array}{rl}
(\varnothing, \{1\}) \models a^1 & \text{by rule 3} \\
(\varnothing, \varnothing) \models b & \text{by rule 2} \\
(\varnothing, \{1\}) \models c^1 & \text{by rule 3}
\end{array}
$$

but, the pattern $\big(b\big)^L \rfloor c^1$ cannot be typed. The only applicable rule is 10, but the premise $N_2' \subseteq N_1'$ is not satisfied because $\{1\} \not\subseteq \varnothing$.

Finally, also the pattern $a^1 \mid b^1 \mid c^1$ (represented in and A.2.(iii)) cannot be typed. In fact

$$
\begin{array}{rl}
(\varnothing, \{1\}) \models a^1 & \text{by rule 3} \\
(\varnothing, \{1\}) \models b^1 & \text{by rule 3} \\
(\varnothing, \{1\}) \models c^1 & \text{by rule 3}
\end{array}
$$

but the only way to type the whole pattern is by applying rule 9 twice. This cannot be done because in the second application of such a rule either the premise $N_1 \cap N_2' \neq \varnothing$ or $N_1' \cap N_2 \neq \varnothing$ is false.

In Fig. A.3 we show other two well–formed examples of interest, representing links between elements standing on different membranes. In both cases we can see how labels can be reused for *links* which "can not see each other", that is, elements bound by the first link can not see elements bound by the second link.

Link 1 of pattern A.3.(i), $\big(a^1\big)^L \rfloor (b^2 \mid e^2) \mid \big(c^1\big)^L \rfloor (f^2 \mid d^2)$, connects two elements standing on two parallel membranes.

$$
\begin{array}{rl}
(\varnothing, \{1\}) \models a^1 & \text{by rule 3} \\
(\varnothing, \{2\}) \models b^2 & \text{by rule 3} \\
(\varnothing, \{2\}) \models e^2 & \text{by rule 3} \\
(\varnothing, \{1\}) \models c^1 & \text{by rule 3} \\
(\varnothing, \{2\}) \models f^2 & \text{by rule 3}
\end{array}
$$

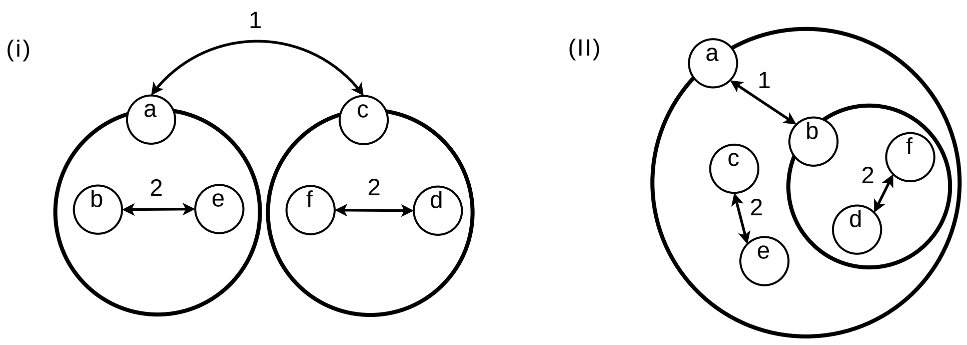

Figure A.3 Example of well–formed patterns in LCLS – links between elements on different membranes and *reuse* of value for links: (i) represents $\left(a^1\right)^L \rfloor (b^2 \,|\, e^2) \,|\, \left(c^1\right)^L \rfloor (f^2 \,|\, d^2)$; (ii) represents $\left(a^1\right)^L \rfloor (c^2 \,|\, e^2 \,|\, \left(b^1\right)^L \rfloor (f^2 \,|\, d^2))$

$(\varnothing, \{2\}) \models d^2$	by rule 3
$(\{2\}, \varnothing) \models b^2 \,|\, e^2$	by rule 9
$(\{2\}, \varnothing) \models f^2 \,|\, d^2$	by rule 9
$(\varnothing, \{1\}) \models \left(a^1\right)^L \rfloor (b^2 \,|\, e^2)$	by rule 10
$(\varnothing, \{1\}) \models \left(c^1\right)^L \rfloor (f^2 \,|\, d^2)$	by rule 10
$(\{1\}, \varnothing) \models \left(a^1\right)^L \rfloor (b^2 \,|\, e^2) \,|\, \left(c^1\right)^L \rfloor (f^2 \,|\, d^2)$	by rule 10

Link 1 of pattern A.3.(ii), $\left(a^1\right)^L \rfloor (c^2 \,|\, e^2 \,|\, \left(b^1\right)^L \rfloor (f^2 \,|\, d^2))$, connects two elements standing on two nested membranes and can be typed as follows:

$(\varnothing, \{1\}) \models a^1$	by rule 3
$(\varnothing, \{2\}) \models c^2$	by rule 3
$(\varnothing, \{2\}) \models e^2$	by rule 3
$(\varnothing, \{1\}) \models b^1$	by rule 3
$(\varnothing, \{2\}) \models f^2$	by rule 3
$(\varnothing, \{2\}) \models d^2$	by rule 3
$(\{2\}, \varnothing) \models c^2 \,|\, e^2$	by rule 9
$(\{2\}, \varnothing) \models f^2 \,|\, d^2$	by rule 9
$(\varnothing, \{1\}) \models \left(b^1\right)^L \rfloor (f^2 \,|\, d^2)$	by rule 10
$(\{2\}, \{1\}) \models c^2 \,|\, e^2 \,|\, \left(b^1\right)^L \rfloor (f^2 \,|\, d^2)$	by rule 9
$(\{1\}, \varnothing) \models \left(a^1\right)^L \rfloor (c^2 \,|\, e^2 \,|\, \left(b^1\right)^L \rfloor (f^2 \,|\, d^2))$	by rule 10

Note that, since the typing algorithm is recursively defined on the structure of patterns, which is always finite, the typing algorithm always terminates.

Once introduced the type–system, the well–formedness of patterns can be defined in the following way:

Definition A.5 (Well–Formedness of Patterns)**.** *A pattern P is* well–formed *if and only if $\models P$ holds.*

It is easy to prove that structural congruence relation preserves well–formedness (see [30]).

Note again that in the following, as we did for well–formedness, even if stating something about links of patterns seems hazarded to us, we retained all the definitions on patterns, as given in [30]. Keep in mind that this will not affect any consideration on terms, since terms are also patterns.

We report some terminology that will be used in the definition of a semantics for LCLS. A well–formed pattern P is *closed* if and only if $(N, \varnothing) \models P$ for some $N \subset \mathbb{N}$ (a closed well–formed pattern contains only complete links, e.g. $a \cdot b^1 \cdot c \,|\, d \cdot x^1$), *open* otherwise (an open well–formed pattern has some unmatched labels at top–level, e.g. $a{\cdot}b^1{\cdot}c^2 \,|\, d{\cdot}x^1$); moreover P is link–free if and only if $(\varnothing, \varnothing) \models P$ (e.g. $a \cdot b \cdot c \,|\, d \cdot x$).

We report the notion of "set of links" of a pattern P, namely the set of labels that occur twice in the top–level compartment of the pattern.

Definition A.6 (Set of Links)**.** *The* set of links *of a pattern P is:*

$$L(P) = \{n | \#(n, L_M(P)) = 2\}$$

where $L_M(P)$ is the multiset of labels *of P, recursively defined as follows:*

1. $L_M(\epsilon) = \varnothing$ 2. $L_M(\nu) = \varnothing$ 3. $L_M(\nu^n) = \{n\}$ 4. $L_M(\widetilde{x}) = \varnothing$
5. $L_M(SP_1 \cdot SP_2) = L_M(SP_1) \cup L_M(SP_2)$
6. $L_M(P_1 \,|\, P_2) = L_M(P_1) \cup L_M(P_2)$
7. $L_M(\big(SP\big)^L \rfloor P) = L_M(SP) \cup (L_M(SP) \cap L_M(P))$
8. $L_M(X) = \varnothing$

where $\nu \in \mathcal{E} \cup EV$, $n \in \mathbb{N}$, P_1, P_2 are any pattern, SP is any sequence pattern.

Note that by its definition, the multiset of labels of a pattern includes the labels in the top–level compartment of such a pattern. In particular, rule 7 deliberately avoids to take into account labels occurring twice in P, in computing the multiset of labels of $\big(SP\big)^L \rfloor P$.

It is easy to see that if P is a well–formed term, there exists $N \subset \mathbb{N}$ such that $(L(P), N) \models P$.

We need to state that two terms which differ only in the natural numbers chosen as their links, represent indeed the same term. The name given to links of a well–formed term, should be a marginal detail, not affecting the semantics of the modeled system. For instance $a^1 \cdot b^1 \cdot c^2$ should be considered equivalent to $a^2 \cdot b^2 \cdot c^1$. To this end, the notions of *alpha*–renaming and α–equivalence are introduced. ,

Definition A.7 (*alpha*–renaming). *Let $\mathcal{A}$ be the set of all total injective functions $\alpha : \mathbb{N} \to \mathbb{N}$. Given $\alpha \in \mathcal{A}$, the* α–renaming *of a LCLS pattern P is the pattern $P\alpha$ obtained by replacing every label n in P by $\alpha(n)$.*

It is possible to show that α–renaming preserves well–formedness (see [30]).

Note that, *reuse* of labels is not essential from the semantics point of view: even if we assume that each label should be unique, we do not affect the meaning of a term. Reusing the same label for links appearing in different compartments (that is, elements bound by the first link can not see elements bound by the second link, as links labeled with 2 in the well–formed pattern $\left(a^1\right)^L \rfloor (b^2 \,|\, e^2) \,|\, \left(c^1\right)^L \rfloor (f^2 \,|\, d^2)$, will make an eventual implementation more efficient in terms of used memory, but on the other side, it makes stating whether two terms differs only in the name of their labels less simpler: if a label is unique, then an α–renaming function, which merely maps each value for labels to a new one, is enough, in order to rename all the labels in a term and obtain the desired semantically equivalent term; while, reusing the same label in different compartments, requires a slight more complicated definition to state which terms have indeed the same meaning. A mindful observer, will notice that renaming links labeled 2 of the previous example, with two different labels, does not affect the semantics of the pattern. Unlike α–renaming functions, the α–equivalence relation takes this aspect into account.

Definition A.8 (α–equivalence). *The α–equivalence relation*[1] $=_\alpha$ *on LCLS patterns is the least equivalence relation which satisfies the following rules:*

$$1. \quad \frac{P_1 \equiv P_2}{P_1 =_\alpha P_2}$$

$$2. \quad \frac{ni \notin L_M(SP_j) \quad i = 1,2 \quad j = 1,2,3}{SP_1 \cdot \nu^{n1} \cdot SP_2 \cdot \mu^{n1} \cdot SP_3 =_\alpha SP_1 \cdot \nu^{n2} \cdot SP_2 \cdot \mu^{n2} \cdot SP_3}$$

$$3. \quad \frac{ni \notin L_M(SP_j) \quad i = 1,2 \quad j = 1,2,3,4}{SP_1 \cdot \nu^{n1} \cdot SP_2 \,|\, SP_3 \cdot \mu^{n1} \cdot SP_4 =_\alpha SP_1 \cdot \nu^{n2} \cdot SP_2 \,|\, SP_3 \cdot \mu^{n2} \cdot SP_4}$$

$$4. \quad \frac{ni \notin L_M(SP_j) \cup L_M(P) \quad i = 1,2 \quad j = 1,2,3,4}{\left(SP_1 \cdot \nu^{n1} \cdot SP_2\right)^L \rfloor (SP_3 \cdot \mu^{n1} \cdot SP_4 \,|\, P) =_\alpha}$$

[1]Remind that an equivalence relation is reflexive, symmetric, and transitive.

$$\left(SP_1 \cdot \nu^{n2} \cdot SP_2\right)^L \rfloor (SP_3 \cdot \mu^{n2} \cdot SP_4 \,|\, P)$$

5. $$\frac{ni \notin L_M(SP_j) \cup L_M(P) \quad i = 1,2 \quad j = 1,2,3,4}{SP_1 \cdot \mu^{n1} \cdot SP_2 \,|\, \left(SP_3 \cdot \nu^{n1} \cdot SP_4\right)^L \rfloor P =_\alpha SP_1 \cdot \mu^{n2} \cdot SP_2 \,|\, \left(SP_3 \cdot \nu^{n2} \cdot SP_4\right)^L \rfloor P}$$

6. $$\frac{ni \notin L_M(SP_j) \cup L_M(P_k) \quad i = 1,2 \quad j = 1,2,3,4 \quad k = 1,2}{\left(SP_1 \cdot \mu^{n1} \cdot SP_2\right)^L \rfloor P_1 \,|\, \left(SP_3 \cdot \nu^{n1} \cdot SP_4\right)^L \rfloor P_2 =_\alpha \left(SP_1 \cdot \mu^{n2} \cdot SP_2\right)^L \rfloor P_1 \,|\, \left(SP_3 \cdot \nu^{n2} \cdot SP_4\right)^L \rfloor P_2}$$

7. $$\frac{ni \notin L_M(SP_j) \cup L_M(P_k) \quad i = 1,2 \quad j = 1,2,3,4 \quad k = 1,2}{\left(SP_1 \cdot \mu^{n1} \cdot SP_2\right)^L \rfloor (P_1 \,|\, \left(SP_3 \cdot \nu^{n1} \cdot SP_4\right)^L \rfloor P_2) =_\alpha \left(SP_1 \cdot \mu^{n2} \cdot SP_2\right)^L \rfloor (P_1 \,|\, \left(SP_3 \cdot \nu^{n2} \cdot SP_4\right)^L \rfloor P_2)}$$

8. $$\frac{\begin{array}{c} P_1 =_\alpha P_2 \quad P_3 =_\alpha P_4 \\ L(P_1) \cap L_M(P_3) = L_M(P_1) \cap L(P_3) = \varnothing \\ L(P_2) \cap L_M(P_4) = L_M(P_2) \cap L(P_4) = \varnothing \end{array}}{P_1 \,|\, P_3 =_\alpha P_2 \,|\, P_4}$$

9. $$\frac{\begin{array}{c} SP_1 =_\alpha SP_2 \quad P_1 =_\alpha P_2 \\ L(SP_1) \cap L_M(P_1) = L_M(SP_1) \cap L(P_1) = \varnothing \\ L(SP_2) \cap L_M(P_2) = L_M(SP_2) \cap L(P_2) = \varnothing \end{array}}{\left(SP_1\right)^L \rfloor P_1 =_\alpha \left(SP_2\right)^L \rfloor P_2}$$

where $\nu, \mu \in \mathcal{E} \cup EV$, $n1, n2 \in \mathbb{N}$, P_1, P_2, P_3, P_4 *are any pattern,* SP_1, SP_2, SP_3, SP_4 *are any sequence pattern.*

Rule 1 states the α–equivalence of congruent patterns;

Rule 2 states that we can rename links in a sequence, provided that the new label does not already occur in the sequence ($ni \notin L_M(SP_j)$);

Rule 3 is similar to rule 2, but in case of link between parallel sequences;

Rule 4 is similar to rule 2, but in case of link between an element on a membrane and an element inside this membrane;

Rule 5 is similar to rule 2, but in case of link between an element on a membrane and an element outside the membrane;

Rule 6 is similar to rule 2, but in case of link between two elements on different parallel membranes;

Rule 7 is similar to rule 2, but in case of link between two elements on nested membranes (one containing the other);

Rule 8 extends the α–equivalence of patterns to the parallel composition of such patterns; when two patterns are composed in parallel, they become part of the same compartment, therefore in stating the α–equivalence of their parallel composition, we must check that the single patterns do not have any link named with the same natural number at their top–level compartment and that the parallel composition does not contain the same label three times, otherwise we would have that their parallel composition would contain the same label four times (note that labels occurring only once in the single patterns, may become a link in their parallel composition);

Rule 9 is the same as rule 8, but in case of looping–and–containment operator.

It is possible to prove that α–equivalence preserves well–formedness of patterns (see [30]).

We can note some differences between the α–equivalence relation and the α–renaming function:

- α–equivalence relation "renames" only complete links, while an α–renaming function can rename each label in a pattern; e.g. a^1 is *alpha*–equivalent only to itself (by rule 1), while it could be renamed in infinite different manners (a^2, a^3, a^4,...) by proper α–renaming functions.
- α–equivalence relation can rename with different labels links appearing in different compartments and initially having the same label, while an α–renaming function would rename them with the same value; e.g. pattern $P_1 = \big(a^1\big)^L \rfloor (c^2 \,|\, e^2 \,|\, \big(b^1\big)^L \rfloor (f^2 \,|\, d^2))$ is α–equivalent to $P_2 = \big(a^1\big)^L \rfloor (c^3 \,|\, e^3 \,|\, \big(b^1\big)^L \rfloor (f^2 \,|\, d^2))$, in fact:

$$
\begin{array}{cr}
c^2 \,|\, e^2 =_\alpha c^3 \,|\, e^3 & \text{by rule 3} \\
\big(b^1\big)^L \rfloor (f^2 \,|\, d^2) \equiv \big(b^1\big)^L \rfloor (f^2 \,|\, d^2) & \text{by rule 1} \\
c^2 \,|\, e^2 \,|\, \big(b^1\big)^L \rfloor (f^2 \,|\, d^2) =_\alpha c^3 \,|\, e^3 \,|\, \big(b^1\big)^L \rfloor (f^2 \,|\, d^2) & \text{by rule 8} \\
P_1 =_\alpha P_2 & \text{by rule 9}
\end{array}
$$

  while an α–renaming function, such that $P_1\alpha = P_2$ does not exist.

Note that, in the following we will apply the concepts of *set of links*, *multiset of links*, *α–equivalence* and *α–renaming* only to terms. However, as said before, we chose to retain the definition on general patterns, following the line of [30], since this will not affect any consideration on terms.

Note that, when we apply an instantiation function σ to a pattern P, we can not merely substitute each occurrence of a variable X with $\sigma(X)$, because we could obtain a non–well–formed term. We report the following examples, taken from [30]: consider the pattern $P_1 = a \cdot \widetilde{x} \,|\, X$ and an instantiation function σ such that $\sigma(\widetilde{x}) = b^1 \cdot c^1$ and $\sigma(X) = d^1 \,|\, e^1$. The application of σ to P_1 would produce the term $P_1\sigma = a \cdot b^1 \cdot c^1 \,|\, d^1 \,|\, e^1$, which is not well–formed. Similarly, consider the pattern $P_2 = a \cdot \widetilde{x} \cdot \widetilde{x}$ and the same instantiation function. We obtain $P_2\sigma = a \cdot b^1 \cdot c^1 \cdot b^1 \cdot c^1$, which is not well–formed.

The solution to this problem is to rename links in the instantiations for variables occurrences, if necessary. The renaming is done finding an α–equivalent term, which will preserve the well–formedness. In this way, concerning the previous example, we could obtain $P_1\sigma = a \cdot b^1 \cdot c^1 \,|\, d^2 \,|\, e^2$ and $P_2\sigma = a \cdot b^1 \cdot c^1 \cdot b^2 \cdot c^2$.

Definition A.9 (Pattern Instantiation)**.** *Given a pattern $P \in \mathcal{P}$ and an instantiation function $\sigma \in \Sigma$, the application of σ to P is a terms $P\sigma$ given by the following inductive definition:*

$$1.\ \ \epsilon\sigma = \epsilon \qquad 2.\ \ a\sigma = a \qquad 3.\ \ a^n\sigma = a^n \qquad 4.\ \ \widetilde{x}\sigma = \sigma(\widetilde{x})$$

$$5.\ \ x\sigma = \sigma(x) \qquad 6.\ \ x^n\sigma = \sigma(x)^n \qquad 7.\ \ X\sigma = \sigma(X)$$

$$8.\ \ \frac{SP_i\sigma =_\alpha S_i \quad L(S_1) \cap L_M(S_2) = L_M(S_1) \cap L(S_2) = \varnothing}{SP_1 \cdot SP_2\ \sigma = S_1 \cdot S_2}$$

$$9.\ \ \frac{P_i\sigma =_\alpha T_i \quad L(T_1) \cap L_M(T_2) = L_M(T_1) \cap L(T_2) = \varnothing}{P_1 \,|\, P_2\ \sigma = T_1 \,|\, T_2}$$

$$10.\ \ \frac{SP\sigma =_\alpha S \quad P\sigma =_\alpha T \quad L(S) \cap L_M(T) = L_M(S) \cap L(T) = \varnothing}{\big(SP\big)^L \rfloor P\ \sigma = \big(S\big)^L \rfloor T}$$

where P_1, P_2, P are any pattern, SP_1, SP_2, SP are any sequence pattern.

Note that given a pattern P, and an instantiation function σ, $P\sigma$ represents indeed a set of *alpha*–equivalent terms.

The definition of rewrite rule is just the same as in CLS.

Definition A.10 (Rewrite Rules)**.** *A rewrite rule is a pair of patterns (P_1, P_2), denoted with $P_1 \mapsto P_2$, where $P_1, P_2 \in \mathcal{P}$, $P_1 \not\equiv \epsilon$ and such that $Var(P_2) \subseteq Var(P_1)$. The infinite set of all the possible rewrite rules is denoted by $\Re$.*

It is finally possible to give an operational semantics of LCLS.

Definition A.11 (Semantics)**.** *Given a set of rewrite rules $\mathcal{R} \subseteq \Re$, such that $\mathcal{R} = \mathcal{R}^{CS} \cup \mathcal{R}^{CU}$ with $\mathcal{R}^{CS} \subset \Re^{CS}$ and $\mathcal{R}^{CU} \subset \Re^{CU}$, the* semantics *of LCLS is the least transition relation $\rightarrow$ on terms closed under $=_\alpha$, and* *satisfying the following inference rules:*

$$\text{(app)}\quad \frac{P_1 \mapsto P_2 \in \mathcal{R} \quad \sigma \in \Sigma \quad \alpha \in \mathcal{A} \qquad (N_1, N') \models P_1\sigma \quad (N_2, N') \models P_2\sigma \quad P_1\sigma \not\equiv \epsilon}{P_1\sigma\alpha \to P_2\sigma\alpha}$$

$$\text{(par)}\quad \frac{T_1 \to T_1' \qquad \models T_1 \,|\, T_2 \qquad \models T_1' \,|\, T_2}{T_1 \,|\, T_2 \to T_1' \,|\, T_2}$$

$$\text{(cont)}\quad \frac{T_1 \to T_1' \qquad \models \left(S\right)^L \rfloor\, T \qquad \models \left(S\right)^L \rfloor\, T'}{\left(S\right)^L \rfloor\, T \to \left(S\right)^L \rfloor\, T'}$$

where the symmetric rule for the parallel composition is omitted.

Rule *(app)* describes the application of a rule: as soon as the two terms, obtained by instantiating the rule patterns, are well–formed and preserve labels occurring only once, we can make a transition between them;

Rule *(par)* propagates the effect of a rewrite rule application to the parallel context, the parallel composition must be well–formed;

Rule *(cont)* is the same as *(par)*, but in case of looping context.

Note that, since the transition relation is closed under $=_\alpha$, if $T_1 \to T_2$ and $T_1 =_\alpha T_1'$ then we also have that $T_1' \to T_2$. Keep in mind that α–equivalence relation does not change labels appearing only once at top–level, this motivates the application of an α–renaming function to the terms obtained by instantiating the rule, in order to permit the application of the same rule to terms which differs only in their labels appearing only once at top–level.

Note that, differently from [30], we allow any kind of fancy rewrite rule to be defined, the only requirement is that its instantiation must be composed by well–formed terms, which do not remove or add a label appearing only once at top–level: the only possible change to this kind of elements is to move them to other positions in the top–level compartment, e.g:

- $a^1 \,|\, b \,|\, \left(c\right)^L \rfloor\, d^1 \cdot e^1 \to b \,|\, \left(c^1\right)^L \rfloor\, d^2 \cdot e^2$ is allowed;
- $a^1 \,|\, b \,|\, \left(c\right)^L \rfloor\, \epsilon \to b \,|\, \left(c\right)^L \rfloor\, a^1$ is not allowed.

Notice that, if we did not allowed reusing of labels in different compartments, we would have not needed to rename the link in the first example.

Bibliography

[1] K. W. Kohn. Molecular interaction map of the mammalian cell cycle control and dna repair systems. *Mol. Biol. Cell*, 10:2703–2734, 1999.

[2] Hiroyuki Kurata, Nana Matoba, and Natsumi Shimizu. Cadlive for constructing a large-scale biochemical network based on a simulation-directed notation and its application to yeast cell cycle. *Nucleic Acids Res.*, 31(14):4071–4084, 2003.

[3] D. L. Cook, J. F. Farley, and S. J. Tapscott. A basis for a visual language for describing, archiving and analyzing functional models of complex biological systems. *Genome Biology*, 2:research00, 2001.

[4] Hiroaki Kitano. A graphical notation for biochemical networks. *Biosilico*, 1:169–176, November 2003.

[5] Kitano H., Funahashi A., Matsuoka Y., and Oda K. Using process diagrams for the graphical representation of biological networks. *Nat Biotechnol*, 23:961–966, 2005.

[6] Kitano H., Funahashi A., Matsuoka Y., and Oda K. A comprehensive pathway map of epidermal growth factor receptor signaling. *Mol Syst Biol.*, 1, 2005.

[7] Kitano H., Funahashi A., and Matsuoka Y. Celldesigner: a process diagram editor for gene-regulatory and biochemical networks. *Biosilico*, 1:159–162, 2003.

[8] R. Maimon and S. Browning. Diagrammatic notation and computational structure of gene networks. In *Proc. 2nd Internat. Conf. on Systems Biology*, pages 330–331, October 2001.

[9] I. Pirson, N. Fortemaison, C. Jacobs, S. Dremier, J. E. Dumont, and C. Maenhaut. The visual display of regulatory information and networks. *Trends in Cell Biology*, 10:404–408, 2000.

[10] K. W. Kohn, M. I. Aladjem, J. N. Weinstein, and Y. Pommier. Molecular interaction maps of bioregulatory networks: a general rubric for systems biology. *Mol Biol Cell*, 17(1):1–13, January 2006.

[11] K. W. Kohn, M. I. Aladjem, S. Kim, J. N. Weinstein, and Y. Pommier. Depicting combinatorial complexity with the molecular interaction map notation. *Mol Syst Biol*, 2, 2006.

[12] K. W. Kohn and M. I. Aladjem. Circuit diagrams for biological networks. *Mol Syst Biol*, 2, 2006.

[13] K. W. Kohn. Molecular interaction maps as information organizers and simulation guides. *Chaos*, 11:84–97, 2001.

[14] K.W. Kohn and V.A. Bohr. Genomic instability and dna repair. *Cancer Handbook*, 1:87–106, 2002.

[15] K. W. Kohn, S. Aladjem, M. I.and Pasa, S. Parodi, and Y. Pommier. Molecular interaction map of mammalian cell cycle control. *Encycl. Human Genome*, 1:457–474, 2003.

[16] K. W. Kohn, J. Riss, O. Aprelikova, J. N. Weinstein, Y. Pommier, and J. C. Barrett. Properties of switch-like bioregulatory networks studied by simulation of the hypoxia response control system. *Mol. Biol. Cell*, 15:3042–3052, 2004.

[17] Mirit I. Aladjem, Stefania Pasa, Silvio Parodi, John N. Weinstein, Yves Pommier, and Kurt W. Kohn. Molecular interaction maps–a diagrammatic graphical language for bioregulatory networks. *Sci. STKE*, 2004(222), 2004.

[18] Y. Pommier, O. Sordet, S. Antony, R. L. Hayward, , and K. W. Kohn. Apoptosis defects and chemotherapy resistance: molecular interaction maps and networks. *Oncogene*, 23:2934–2949, 2004.

[19] K. W. Kohn and Y. Pommier. Molecular interaction map of the p53 and mdm2 logic elements that switch on the response of p53 to dna damage. *Biochem. Biophys. Res. Commun.*, 331:816–827, 2005.

[20] H. Matsuno, A. Doi, M. Nagasaki, and S. Miyano. Hybrid petri net representation of gene regulatory network. *Pacific Symposium on Biocomputing*, pages 341–352, 2000.

[21] M. Curti, P. Degano, C. Priami, and C.T. Baldari. Modelling biochemical pathways through enhanced pi-calculus. *Theoretical Computer Science*, 325(1):111–140, 2004.

[22] A. Regev, W. Silverman, and E.Y. Shapiro. Representation and simulation of biochemical processes using the pi-calculus process algebra. In *Pacific Symposium on Biocomputing*, pages 459–470. World Scientific Press, 2001.

[23] L. Cardelli. Brane calculi. interactions of biological membranes. *Computational Methods in Systems Biology (CMSB'04)*, pages 257–280, 2005.

[24] C. Priami and P. Quaglia. Beta binders for biological interactions. *Computational Methods in Systems Biology (CMSB'04)*, pages 20–33, 2005.

[25] R. Barbuti, S. Cataudella, A. Maggiolo-Schettini, P. Milazzo, and A. Troina. A probabilistic model for molecular systems. *Fundamenta Informaticae*, 67:13–27, 2005.

[26] G. Păun. Membrane computing. an introduction. 2002.

[27] Roberto Barbuti, Andrea Maggiolo-Schettini, Paolo Milazzo, and Angelo Troina. The calculus of looping sequences for modeling biological membranes. In Springer, editor, *Membrane Computing*, volume LNCS 4860, pages 54–76. Springer, June 2007.

[28] Roberto Barbuti, Andrea Maggiolo-Schettini, Paolo Milazzo, Paolo Tiberi, and Angelo Troina. Stochastic cls for the modeling and simulation of biological systems. Draft, 2007.

[29] Paolo Milazzo. *Qualitative and Quantitative Formal Modeling of Biological Systems.* PhD thesis, University of Pisa, April 2007.

[30] Roberto Barbuti, Andrea Maggiolo-Schettini, and Paolo Milazzo. A formalism to model protein interaction at the domain level. Draft, 2007.

[31] D. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81:2340–2361, 1977.

[32] Roberto Barbuti, Andrea Maggiolo-Schettini, Paolo Milazzo, and Angelo Troina. A calculus of looping sequences for modelling microbiological systems. *Fundamenta Informaticae*, 72(1-3):21–35, 2006.

[33] Roberto Barbuti, Andrea Maggiolo-Schettini, and Paolo Milazzo. Extending the calculus of looping sequences to model protein interaction at the domain level. In *Bioinformatics Research and Applications*, volume 4463, pages 638–649. Springer, May 2007.

[34] F. Ciocchetta, C. Priami, and P. Quaglia. Modeling kohn interaction maps with beta-binders: An example. *Transactions on Computational Systems Biology, (III)*, volume 3737:22–48, 2005.

[35] Simone Fonda. Simulazione combinatoria a vincoli di mappe di interazione molecolare. Master's thesis, University of Pisa, 2007.

[36] Vincent Danos and Cosimo Laneve. Formal molecular biology. *Theor. Comput. Sci.*, 325(1):69–110, 2004.

[37] Nathalie Chabrier-Rivier, Marc Chiaverini, Vincent Danos, François Fages, and Vincent Schächter. Modeling and querying biomolecular interaction networks. *Theor. Comput. Sci.*, 325(1):25–44, 2004.

[38] R. Milner. Communicating and mobile systems: the π-calculus. 1999.

[39] L. Bortolussi. Stochastic concurrent constraint programming. *In Proceedings of 4th International Workshop on Quantitative Aspects of Programming Languages*, 164:6580, 2006.