Multi-Scale Modelling of Biological Systems in Process Algebra

Andrea Degasperi

Submitted in fulfilment of the requirements for the Degree of
Doctor of Philosophy

School of Computing Science
College of Science and Engineering
University of Glasgow

July 2011

Abstract

There is a growing interest in combining different levels of detail of biological phenomena into unique multi-scale models that represent both biochemical details and higher order structures such as cells, tissues or organs.

The state of the art of multi-scale models presents a variety of approaches often tailored around specific problems and composed of a combination of mathematical techniques. As a result, these models are difficult to build, compose, compare and analyse.

In this thesis we identify process algebra as an ideal formalism to multi-scale modelling of biological systems.

Building on an investigation of existing process algebras, we define process algebra with hooks (PAH), designed to be a *middle-out* approach to multi-scale modelling. The distinctive features of PAH are: the presence of two synchronisation operators, distinguishing interactions within and between scales, and composed actions, representing events that occur at multiple scales. A stochastic semantics is provided, based on functional rates derived from kinetic laws. A parametric version of the algebra ensures that a model description is compact.

This new formalism allows for: unambiguous definition of scales as processes and interactions within and between scales as actions, compositionality between scales using a novel vertical cooperation operator and compositionality within scales using a traditional cooperation operator, and relating models and their behaviour using equivalence relations that can focus on specified scales.

Finally, we apply PAH to define, compose and relate models of pattern formation and tissue growth, highlighting the benefits of the approach.

To my parents Giancarlo and Marisa and to my brother Matteo.
To Melita, Daniela, Pietro, Cristina, Chiara, Anna, Andrea,
Luca, Stefano, Maurizio, Martin, Karin,
Rosetta, Giorgio, Ida, Mariano,
Rosetta, Silvio, Silvana, Willy,
Rina, Livio, Lina, Giovanni.
Always in my heart wherever I am.

Acknowledgements

I would like to thank Prof. Muffy Calder for the guidance and support she has given me throughout these years. It has been great to have her always on my side to help me improve my research skills and to encourage me to pursue all my ideas.

I would like to thank also Dr. Federica Ciocchetta, a talented researcher and a friend, for the fun we had working together.

Finally I would like to thank Oberdan and Michele, who made me feel that Italy and Trentino were not that far away from Glasgow, and my family, in particular my parents, who always supported me and my decisions, making me feel that I could accomplish anything I wanted.

Declaration

A preliminary version of the process algebra introduced in Section 5.2 has been published (Degasperi and Calder, 2010) under the supervision of Prof. Muffy Calder.

The process algebra introduced in Section 5.2, the definition of isomorphism and related properties in Section 6.2.1 and the case study in Section 7.3 have been published (Degasperi and Calder, 2011) under the supervision of Prof. Muffy Calder.

All the work reported in this thesis has been performed by myself, unless specifically stated otherwise.

Andrea Degasperi

July 2011

Abbreviations

SBML	-	systems biology markup language
ODE	-	ordinary differential equation
PDE	-	partial differential equation
CME	-	chemical master equation
SSA	-	stochastic simulation algorithm
CTMC	-	continuous time Markov chain
CA	-	cellular automata
CCS	-	calculus of communicating systems
CSP	-	communicating sequential processes
PEPA	-	performance evaluation process algebra
EMPA	-	extended Markovian process algebra
SPA	-	simple process algebra
sSPA	-	stochastic simple process algebra
pSPA	-	parametric simple process algebra
PAwP	-	process algebra with priorities
sPAwP	-	stochastic process algebra with priorities
PAH	-	process algebra with hooks
sPAH	-	stochastic process algebra with hooks
psPAH	-	parametric stochastic process algebra with hooks

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Systems Biology and traditional modelling techniques. Systems Biology (Kitano, 2002) is an emerging discipline that aims to improve our understanding of the dynamics of biological processes with the aid of mathematical models. As our knowledge about the mechanics and the complexity of biological phenomena increases, predictive models become necessary to validate understanding and generate new hypotheses.

The level of detail at which biological processes are most commonly modelled is biochemical reactions, using mathematical approaches such as ordinary differential equations (ODE) and stochastic processes (Klipp et al., 2005). These approaches are used to represent the change in time of the concentration or number of molecules involved in the reactions, under the assumption that they are well mixed and at constant temperature. If more complex phenomena are considered, such as organogenesis (organ development) or tissue growth, other approaches are employed to represent diffusion of molecules, using partial differential equations (PDE) (Meinhardt, 2008), or higher order structures such as cells or tissue, using cellular automata (CA) (Ermentrout and Keshet, 1993) or other agent based techniques. With PDEs the change of concentration of molecules at different positions in space can be modelled and boundaries for the diffusion can be defined. With CA, individual cells and their behaviour, such as movement in space, can be modelled.

Formal Methods for Systems Biology. Alongside the mentioned modelling approaches, descriptive languages, e.g. SBML (Hucka et al., 2003), and graphical notations, e.g. Kitano Map (Kitano, 2003), have been developed to help writing,

maintaining and sharing models. This is achieved with unambiguous and descriptive definitions of components and interactions within a model. In addition, formalisms from the field of computer science have been proposed not only to provide an unambiguous definition of biological phenomena, but also to improve the overall modelling approach. They are characterised by being *executable*, i.e. they can produce behaviour according to one or more associated semantics. Most importantly, both the syntax and semantics of these formalisms are mathematically well defined, creating a mathematical framework where formal reasoning between syntax and semantics is possible. Some of the most successful formalisms are process algebras and other calculi (Bortolussi, 2006; Ciocchetta and Hillston, 2009; Hillston, 1996; Priami and Quaglia, 2005), rewriting rules (Blinov et al., 2004; Danos et al., 2007) or programming languages (Calzone et al., 2006; Pedersen and Plotkin, 2008).

Process Algebra for Systems Biology. Process algebras are a family of calculi developed to represent and analyse formally the behaviour of concurrent systems, such as programs on a computer or computers in a network (Hoare, 1985; Milner, 1989). They have been shown to be one of the most promising approaches to the formalisation of biological systems, because of the deep analogies that exist between concurrent agent interactions and biochemical reactions (Regev et al., 2001). In particular, process algebra provides:

- a formalisation of biological systems, where biological entities are represented by processes and biological events are represented by actions that the processes can perform asynchronously or synchronously. Synchronisation can be *binary*, where exactly two processes participate to an action, or *multi-way*, where two or more processes participate to an action;

- compositionality, i.e. the possibility of constructing a system as the sum of its constituent components. This is represented as a *cooperation* or *parallel composition* of processes;

- well established techniques to reason about behaviour. In particular, a theory of relations based on behaviour, that allows systems to be compared or part of a system to be abstracted with other parts that are behaviourally equivalent;

- multiple semantics can be derived automatically from a model definition, which allow for complementary analysis techniques such as ordinary differential equations, stochastic simulation or continuous time Markov chains.

The development and application of process algebra for biology has usually been aimed at modelling biochemical reactions and compartments. Specifically, it has involved the modelling of a single level of detail, whether this is biochemistry (Calder et al., 2006a; Ciocchetta and Hillston, 2009; Priami and Quaglia, 2005; Priami et al., 2001) or membrane (Cardelli, 2005). In an exception, multiple levels of detail are modelled with a single process algebra in Bioambients (Regev et al., 2004), where biochemistry and modifications of membranes, and so spatial organisation, are considered.

Multi-scale modelling. More recently there has been a growing interest in combining different levels of detail of biological phenomena into single multi-scale models that represent both biochemical details and higher order structures. This is a necessary step to achieve a complete understanding of the emerging behaviour in a complex biological phenomenon. Model construction follows mainly two approaches: *bottom-up* and *top-down*. The former begins from identifying elementary parts, such as molecules, and aims at explaining more complex phenomena as the emergent behaviour of its components. The latter begins instead from reproducing observed phenomena and then adds internal details, attempting to recreate governing mechanisms. Different mathematical approaches are often considered for different scales and integrated into a multi-scale model tailored around a specific biological problem (Dada and Mendes, 2011; Walker and Southgate, 2009; Walker et al., 2008). As a consequence, composition and comparison of two multi-scale models is often very difficult.

It has been proposed (Noble, 2006) that new, more flexible modelling techniques should allow for a *middle-out* approach. This means that one begins studying, and so modelling, a biological phenomenon from any level of detail or spatial scale and, in a second stage, extending its study and so its model either up scale, integrating with other components, or down scale, adding more internal details. To our knowledge, no formal approach has been proposed that specifically addresses the problem of integrating multiple scales under the same mathematical framework and that has the flexibility of treating different scales as the same formal objects.

Process algebra as 'middle-out' approach. In this thesis we propose that process algebra is a perfect candidate as a middle-out approach for multi-scale modelling. In particular, its natural support of compositionality and its abstraction mechanisms can provide the required flexibility that writing and composing multi-scale models require. This leads us to our thesis statement.

1.1 Thesis Statement

There is currently a need for a flexible and compositional modelling approach that supports the integration of multiple scales, to aid the understanding of complex biological phenomena such as organogenesis and tumour growth.

We propose that process algebra is a perfect candidate, because of its natural support of compositionality and its abstraction mechanisms.

We demonstrate this by developing and applying a process algebra dedicated to the multi-scale modelling of biological systems, after having explored the limitations of current process algebraic approaches.

1.2 The Choice of Multi-Way Synchronisation

In this thesis we consider mainly process algebras with multi-way synchronisation. Our choice of multi-way over binary synchronisation is motivated as follows:

- it allows one to model biochemical reactions with any number of reactants and products with a single action, and so atomically (Calder and Hillston, 2009). Moreover, a rate based on one of a variety of kinetic laws (Segel, 1993) can be associated to that action. In general, kinetic laws approximate sequences of reactions and are employed when it is difficult to measure rates of some of those reactions in biological experiments;
- it is more amenable for a multi-scale scenario, where multiple scales can be affected by the same event at the same time.

1.3 Contributions

The main contributions of this thesis are:

- the definition of *process algebra with hooks*, a novel process algebra designed for multi-scale modelling of biological systems. Its main features are: explicit modelling of scales and interactions within and between scales; use of *composed* actions in a multi-way synchronisation setting; a *vertical* cooperation operator in addition to the standard cooperation operator for composition of processes; a stochastic semantics based on functional rates;

- the definition of a functional rate semantics for process algebras based on biological principles, where actions can be rated only if *closed*, i.e. only if all the expected participants to that action synchronise;

- an investigation of the use of a simple process algebra and a process algebra with action priorities in the multi-scale scenario. This investigation highlights drawbacks that are addressed by process algebra with hooks;

- the definition of three congruence relations to relate and substitute process algebra with hooks processes. They relate processes by their structure (isomorphism), by their structure with focus on a specified scale ($(\mathcal{T}, \Gamma)$-isomorphism) and by their spatial and temporal behaviour at a specified scale (Markovian $(\mathcal{T}, \Gamma)$-bisimulation). The proof of congruence for $(\mathcal{T}, \Gamma)$-isomorphism and Markovian $(\mathcal{T}, \Gamma)$-bisimulation is possible because of the concept of closed actions introduced with our definition of functional rates;

- the illustration of use of process algebra with hooks to model, simulate and relate multi-scale models of pattern formation and tissue growth.

1.4 Publications

Investigation of the thesis and related topics led to the following publications:

- A. Degasperi and M. Calder. Multi-Scale Modelling of Biological Systems in Process Algebra with Multi-Way Synchronisation. *CMSB 2011, to appear in ACM Digital Library*, 2011;

- A. Degasperi and M. Calder. Process Algebra with Hooks for Models of Pattern Formation. *CS2Bio2010, ENTCS 268*, pages 31-47, Elsevier, 2010;

- A. Degasperi and M. Calder. Relating PDEs in Cylindrical Coordinates and CTMCs with Levels of Concentration. *CS2Bio2010, ENTCS 268*, pages 49-59, Elsevier, 2010;
- A. Degasperi and M. Calder. On the Formalisation of Gradient Diffusion Models of Biological Systems. *PASTA Workshop 2009*, unreviewed, 2009;
- F. Ciocchetta, A. Degasperi, J. Hillston, M. Calder. Some investigations concerning the CTMC and the ODE model derived from Bio-PEPA. *FBTC 2008, ENTCS 229(1)*, pages 145-163, Elsevier, 2009.

1.5 Thesis Outline

The thesis is organised as follows. After discussing background material in Chapter 2, we investigate the use of a simple process algebra with multi-way synchronisation (SPA) to model a single spatial scale in Chapter 3. In Chapter 4 we move the focus on to multi-scale modelling showing how SPA is not suited to model desired inter-scale interactions and how a process algebra with priorities (PAwP) can be used as well as the drawbacks of this approach. Building on the previous chapters, we propose process algebra with hooks (PAH) in Chapter 5 which presents the advantages of PAwP with respect to SPA, without its drawbacks. In Chapter 6 we continue the characterisation of PAH introducing three equivalence relations, while in Chapter 7 we apply a parametric stochastic version of PAH to the modelling of two case studies. Conclusions and future work are in Chapter 8.

We now present a more detailed overview of the thesis.

In Chapter 2 we cover background material, from useful biological concepts in Section 2.1, to a survey of traditional modelling approaches in Section 2.2, and a discussion of formal methods for systems biology with focus on process algebra in Section 2.3.

In Chapter 3 we show how a simple process algebra with multi-way synchronisation (SPA) can be used to model a single spatial scale. First, we introduce a non stochastic version of the semantics (Section 3.1) and propose examples of modelling biochemistry and tissue growth (Sections 3.1.1 and 3.1.2). Second, we consider a stochastic semantics for simple process algebra (sSPA) based on functional rates (Section 3.2). An example of the application of stochastic simple process algebra is in Sections 3.2.5. To conclude the chapter on modelling a

single scale, we propose a parametric process algebra (pSPA) that makes model definition more compact (Section 3.3). Examples are given in Sections 3.3.1 and 3.3.2.

In Chapter 4 we introduce the concept of interactions between scales (Section 4.1) and discuss how SPA is not ideal to model such interactions (Section 4.1.1). Then, we introduce a process algebra with priority of actions (PAwP, in Section 4.2). Actions with low priority are considered local and represent the behaviour of a single scale. Actions with high priority are considered inter-scale interrupts, i.e. signals operating between scales. We illustrate how this algebra can be employed successfully to the multi scale modelling of biological systems with examples (Sections 4.2.3 and 4.2.2). Finally, we highlight some drawbacks of the use of action priorities: the creation of additional, intermediate, biologically meaningless states; the lack of control by the modeller when multiple inter-scale signals happen at the same time; the lack of explicit syntactic elements that could unambiguously represent scales and actions operating within and between scales and that could improve the overall compositionality of the algebra.

In Chapter 5 we introduce process algebra with hooks (PAH, in Section 5.1), which is designed for multi scale modelling of biological systems and which addresses the drawbacks identified in PAwP. In particular, instead of using separate actions for intra and inter-scale interactions, *composed* actions are used. This means that a single composed action can perform both interactions within and between scales. Examples are presented in Sections 5.1.3 and 5.1.2. A comparison between PAwP and PAH is given in Section 5.1.4. Stochastic process algebra with hooks (sPAH) is introduced in Section 5.2 with examples in Sections 5.2.4 and 5.2.3.

In Chapter 6 we continue the characterisation of sPAH, defining congruences on processes, which can relate processes with equivalent behaviour at a specified scale, abstracting away as much as possible from other scales. Although at a certain scale the behaviour of biological systems can be different, e.g. different biochemical networks are present, at a higher or lower scale behaviour could be analogous under a certain notion of equivalence, e.g. in both cases cells proliferate and die at the same rate. Equivalence relations can also be used to substitute parts within a model with equivalent and possibly less complex alternatives (Figure 1.1).

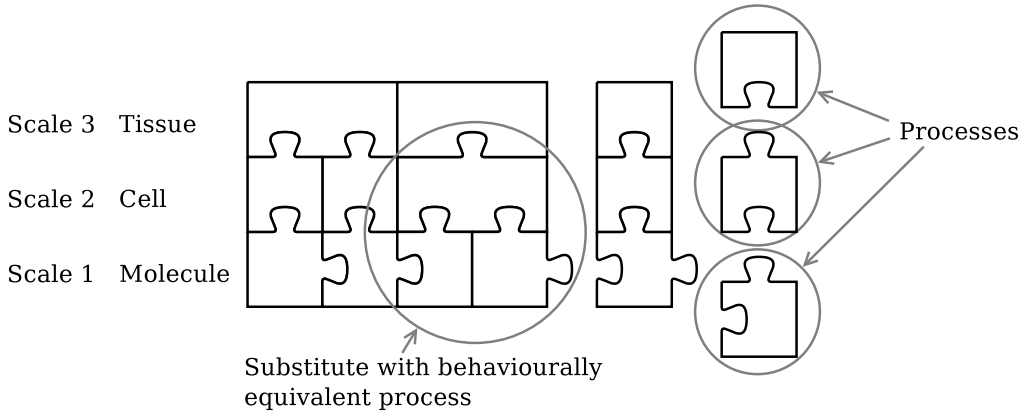


Figure 1.1: Jigsaw representation of compositionality and behaviour abstraction in a multi-scale model.

In Chapter 7 we introduce a parametric version of sPAH, called psPAH, and illustrate the use of the algebra in two case studies: multi-scale models of pattern formation (Section 7.2) and tissue growth (Section 7.3). In particular, we show how to define, simulate and relate models in psPAH.

In Chapter 8 we present our conclusions and future work related to this thesis.

Finally, in Appendix A we define for completeness a stochastic version of PAwP, using our approach to functional rates, and in Appendix B we give a complete definition of the multi-scale model of tissue growth of Section 7.3.

Chapter 2

Background

In this chapter we survey useful biological concepts in Section 2.1, traditional modelling approaches in Section 2.2 and discuss formal methods for systems biology with focus on process algebra in Section 2.3.

2.1 Biological Concepts

In this section we cover concepts useful to the understanding of the biology in the thesis. Our references for this section are (Nelson and Cox, 2004) and (Klipp et al., 2005).

2.1.1 The Complexity of Organisms

Every organism, whether it is an animal, a plant or a bacterium, consists of biochemical molecules, which participate in complex interactions and collective behaviour. These molecules have highly specific functions and can be organised in higher order structures, such as membranes, cells, tissues or organs. Physical forces and chemical reactions allow organisms to function as dynamic entities, able to sense the environment they are in and respond accordingly. In this introduction, we discuss some of the principles that allow cells to make decisions, with focus on pattern formation in the development of organisms.

We begin by explaining what proteins are and how they can interact to create metabolic and signalling networks. Then we overview mechanisms of cell differentiation and memory.

2.1.2 Proteins

In order to explain what proteins are, we briefly introduce the *central dogma* of molecular biology. DNA (deoxyribonucleic acid) is the molecule, in every cell, that contains information about how to construct (synthesise) proteins. This information is coded as a sequence of bases: adenine, thymine, guanine and cytosine. When a protein needs to be built, a process called *transcription* copies the necessary sequence of bases from the DNA to a strand of RNA (ribonucleic acid). Then, during *translation*, the RNA binds to a molecule called ribosome and the information present in the RNA is used to construct a chain of amino acids. After a chain of amino acids is formed, this chain folds, thanks to bonds and forces acting on it, leading to the final shape of the protein.

A *gene* is a sequence of DNA which encodes one or more proteins or a strand of RNA that has a function in the cell or organism. A gene is said to be *expressed* if its sequence of DNA is transcribed and, possibly, translated into a protein.

Proteins fulfil numerous functions in the cell, from being just part of the cellular structure to having roles in the metabolism of the cell or in the delivery of signals. The main characteristic of proteins that enables them to have so many different functions is their ability to bind to other molecules specifically and tightly. The regions in the protein where other molecules may bind are called *binding sites*. These regions are defined by their shape and by the chemical properties that surround them, allowing only very specific molecules to bind. Proteins can also bind to other proteins or be integrated into membranes. When a protein binds to another molecule, it can also change some of its properties and abilities to bind.

Enzymes. An enzyme is a protein whose role is to catalyse, i.e. to accelerate, a biochemical reaction. Enzymes allow reactions that are normally unfavourable in nature to take place, lowering their activation energy. We will call *reactants* the molecules that take part in catalysed reactions and *products* the molecules that are generated. Usually enzymes only catalyse very specific reactions.

2.1.3 Metabolic and Signalling Pathways

The metabolism of a cell is a highly organised process, that involves thousands of reactions that are catalysed by enzymes and whose ultimate goal is to provide everything the cell needs to survive and reproduce. Metabolism provides

energy and material for building and maintaining the cell. Metabolic pathways are networks of biochemical interactions that involve mainly mass and energy transfer.

On the other hand, a signalling pathway is a sequence of biochemical interactions that leads to the transmission of external signals from outside to inside the cell and to the movement of information inside the cell. Examples of signals are hormones, pheromones, heat, cold, light or even the appearance or concentration change of substances such as glucose or potassium or calcium ions. The interpretation of these external signals triggers the cell response.

2.1.4 Biological Mechanisms of Cell Differentiation and Decision Making

The *genome*, i.e. the set of all genes of an organism, is normally identical in every cell. Cell differentiation, and so specialisation of function, is achieved by selecting different genes to be expressed in individual cells, while the genetic information contained in all of them is mostly identical. Gene selection controls four essential processes of a cell: cell proliferation, cell specialisation, cell interactions and cell movements.

Many biological processes are transient, i.e. changes in gene expression are temporary. For example, a response to an external signal can activate genes as a response. When the signal is gone, the response ceases.

A stable choice of gene expressions is possible because of *cell memory*. Which genes are expressed depends on the past, along with the present environment. *Memory* is essential for the creation of organised tissues and for the *stable maintenance* of cell specialisation.

Although other mechanisms are possible, cell differentiation is mainly achieved by *sensing* concentration levels of specific proteins. Even a single protein, present in high concentration, can activate entire pathways and transcription circuits, deciding irreversibly the fate of the cell it is in. Multiple thresholds (for example high, medium and low concentration) are not uncommon.

Thus, if the concentration of a protein in a cell can determine its differentiation, two originally identical cells with different fates must have reached a different concentration level for that protein at a key moment in time, when the selection took place. How this different concentration level arises is probably the

most important question addressed by the field of *Developmental Biology*. In general, this phenomenon involves a notion of spatial location.

Using memory, cells can remember their position, referred to as *positional value*. During organism development, the memorisation of position is often an intermediate step between non-specialised and specialised cells.

A group of cells can be influenced by a signal coming from neighbouring cells, called an *inductive signal*, driving one or more of the members of the group into a different developmental pathway. This process is called *inductive interaction* and it consists of a signal limited in time and space. An inductive signal can be long range, e.g. highly diffusible molecules, or short range, e.g. cell-to-cell interactions. Inductive signal molecules are often referred to as *morphogens*.

2.2 Traditional Modelling Methods

Biological interactions can be modelled and studied at different levels of detail. It is possible to concentrate on the properties of individual reactions as well as studying the system as a whole. In this section, we consider models of biochemical reactions, as this is the most popular level of detail.

The main mathematical approaches to quantitative analysis we discuss in this section are: *Ordinary Differential Equations* (ODEs), *Stochastic Simulation Algorithm* (SSA) based on the *Chemical Master Equation* (CME), *Continuous Markov Chain with levels* (CTMCs with levels) and *Reaction-Diffusion Equations* (RDE).

2.2.1 ODE: The Law of Mass Action

ODEs are the most common way of modelling chemical or biochemical interactions. They express the rate of change in time of the concentration of the participants of a biochemical reaction in an environment where molecules are well-mixed. Each biochemical reaction is associated with a rate of change, called the *velocity* of the reaction. Velocities are usually dependent on the concentration of the molecules that are involved in the modelled reaction. Often, a velocity is expressed in terms of a *kinetic law*, i.e. an equation that expresses the dynamics of multiple biochemical interactions at a time. A kinetic law may take into

account which molecules can be measured in a biological experiment in order to determine the value of constant parameters (Segel, 1993).

The key characteristic of the ODE approach is the fact that it shows the most likely behaviour of the system, assuming continuous concentrations, i.e. an infinite number of interacting molecules.

A common kinetic law is the law of mass action, introduced in the 19th century (Guldberg and Waage, 1879). It states that the rate at which a species is produced or consumed by a reaction is proportional to the amount of reactants and the *stoichiometry*, i.e. how many copies of a reactant are involved in the reaction. For example, the velocity of the reaction

$$\mathbf{S}_1 + \mathbf{S}_2 \rightleftarrows 2\mathbf{P}$$

can be formulated as

$$v = v_+ - v_- = k_+ \cdot [\mathbf{S}_1] \cdot [\mathbf{S}_2] - k_- \cdot [\mathbf{P}]^2$$

where $\mathbf{S}_1$, $\mathbf{S}_2$ and $\mathbf{P}$ are molecular species, v is the velocity, v_+ is the velocity of only the forward reaction, v_- is the velocity of the backward reaction and k_+ and k_- are the proportionality factors, called *kinetics* or *rate constants*. The symbol $[\cdot]$ denotes the concentration of the species, usually expressed in moles per litre (mol/L) or molar (M). We use bold-capital font for molecules. The dynamics of the concentrations of the species can be described by *Ordinary Differential Equations* (ODEs) for example the equations for the reactions above are given by:

$$\begin{aligned} \frac{d[\mathbf{S}_1]}{dt} &= \frac{d[\mathbf{S}_2]}{dt} = -v \\ \frac{d[\mathbf{P}]}{dt} &= 2v \end{aligned}$$

The value of the concentrations of $\mathbf{S}_1$, $\mathbf{S}_2$ and $\mathbf{P}$ through time are obtained by integration of these ODEs.

2.2.2 Generalised Mass Action

In this section we generalise and formalise the concepts we introduced in the previous section.

Modelling intracellular dynamics in a quantitative way is concerned with the estimation through time of the number of molecules of n different species $\mathbf{S}_1, \ldots, \mathbf{S}_n$, which can interact according to m biochemical reactions R_j. In general, a biochemical reaction R_j can be formalised as follows:

$$R_j : \kappa_{j1}\mathbf{S}_{p(j,1)} + \kappa_{j2}\mathbf{S}_{p(j,2)} + ... + \kappa_{jL_j}\mathbf{S}_{p(j,L_j)} \xrightarrow{k_j} \kappa_{jL_j+1}\mathbf{S}_{p(j,L_j+1)} + ... + \kappa_{jT_j}\mathbf{S}_{p(j,T_j)}$$

where L_j is the number of reactants and T_j is the number of reactants *and* products in R_j, κ_{jz} is the stoichiometric coefficient of the reactant species $\mathbf{S}_{p(j,z)}$, $K_j{=}\sum_{z=1}^{L_j} \kappa_{jz}$ denotes the molecularity of the reaction R_j and the index $p(j,z)$ selects those $\mathbf{S}_i$ participating in R_j. The stoichiometric coefficient indicates how many copies of a species participate to a reaction.

Assuming a constant temperature and that diffusion in the cell is fast, so that we can assume a homogeneously distributed mixture in a fixed volume V, the *General Mass Action* (GMA) model of the system can be defined by n ordinary differential equations (ODEs) as follows:

$$\frac{d[\mathbf{S}_i]}{dt} = \sum_{j=1}^{m} d_{ji} k_j \prod_{z=1}^{L_j} [\mathbf{S}_{p(j,z)}]^{\kappa_{jz}} \qquad i = 1, 2, ..., n \tag{2.1}$$

where the k_js are rate constants, d_{ji} denotes the change in molecules of $\mathbf{S}_i$ resulting from a single R_j reaction and m is the number of reactions. Symbol $[\mathbf{S}_i]$ is the concentration of the species $\mathbf{S}_i$.

2.2.3 Michaelis-Menten kinetics

Other kinetic laws can be obtained adding further assumptions to a set of Mass Action equations. An example is the Michaelis-Menten kinetics, a model of enzymatic reactions that is well established in the field of systems biology (Briggs and Haldane, 1925):

$$\mathbf{E} + \mathbf{S} \overset{k_1, k_{-1}}{\longleftrightarrow} \mathbf{ES} \xrightarrow{k_2} \mathbf{E} + \mathbf{P}$$

where $\mathbf{E}$ is the enzyme, $\mathbf{S}$ the substrate, $\mathbf{ES}$ the temporary enzyme-substrate complex and $\mathbf{P}$ is the product of the reaction. Characteristics of this model are that the process is considered irreversible, i.e. the product cannot become a

substrate, and the enzyme is not affected by the reactions and can be used again after it leaves the substrate or the product.

The ODEs of the model, according to the GMA law, are the following:

$$\frac{d[\mathbf{S}]}{dt} = -k_1 \cdot [\mathbf{E}] \cdot [\mathbf{S}] + k_{-1} \cdot [\mathbf{ES}]$$

$$\frac{d[\mathbf{ES}]}{dt} = k_1 \cdot [\mathbf{E}] \cdot [\mathbf{S}] - (k_{-1} + k_2) \cdot [\mathbf{ES}]$$

$$\frac{d[\mathbf{E}]}{dt} = -k_1 \cdot [\mathbf{E}] \cdot [\mathbf{S}] + (k_{-1} + k_2) \cdot [\mathbf{ES}]$$

$$\frac{d[\mathbf{P}]}{dt} = k_2 \cdot [\mathbf{ES}]$$

This system of ODEs can be simplified using further assumptions. One of these is that we consider the conversion of **E** and **S** into **ES** and vice versa to be much faster than the decomposition of **ES** into **E** and **P** ($k_1, k_{-1} \gg k_2$, the *quasi equilibrium* assumption). The other assumption is that during the course of the reactions a state is reached where the concentration of **ES** remains constant. This is called the *quasi steady-state* assumption, due to the fact that we consider the concentrations of the intermediates (**ES**) to reach equilibrium much faster than those of the product and substrate. Using these assumptions and some simple manipulation, one can obtain the following simplified velocity for the above enzymatic reaction:

$$\frac{d[\mathbf{P}]}{dt} = k_2[\mathbf{E}_{tot}]\frac{[\mathbf{S}]}{[\mathbf{S}] + K_m} = \frac{V_{max}[\mathbf{S}]}{[\mathbf{S}] + K_m}$$

where $[\mathbf{E}_{tot}] = [\mathbf{ES}] + [\mathbf{E}]$, V_{max} (also written k^{cat}) is the maximum velocity of the production of P, given by $k_2[\mathbf{E}_{tot}]$ and $K_m = (k_{-1} + k_2)/k_1$ is called the *Michaelis constant*. The parameters V_{max} and K_m can be easily estimated with few biological experiments.

2.2.4 Chemical Master Equation

CME based approaches were introduced in order to take into account the stochastic effect due to the probability of molecule collisions, especially when the number of molecules of a species reduces to a few units (McQuarrie, 1967). In the CME,

each molecule of the system is modelled and has a probability to react, thus allowing modelling of likelihood of states of the system. A state is a snapshot of the number of molecules for each species at a given time. This is at the price of a sometimes problematic computational complexity.

In CME based approaches we wish to determine for each molecular species $\mathbf{S}_i$ the probability $P(\#\mathbf{S}_i(t) = s_i)$ that at time t there are s_i molecules (with $\#\mathbf{S}_i$ denoting the number of molecules of the species $\mathbf{S}_i$). For n molecular species, let $\mathbf{s} \in \mathbb{N}^n$ denote the n dimensional state vector. The vectors $\mathbf{d}_j \in \mathbb{Z}^n$ are the step changes occurring for elementary reactions indexed by j. If $\mathbf{S}$ is an n dimensional variable, we write $P(\#\mathbf{S} = \mathbf{s})$ as $P_{\mathbf{s}}(t)$. In order to describe the changes in random variable $\mathbf{S}$, we consider the following two state transitions:

$$\mathbf{s} - \mathbf{d}_j \xrightarrow{a_j(\mathbf{s}-\mathbf{d}_j)} \mathbf{s}$$
$$\mathbf{s} \xrightarrow{a_j(\mathbf{s})} \mathbf{s} + \mathbf{d}_j$$

The first denotes a transition from another state to the state $\mathbf{s}$; the second denotes moving away from the state $\mathbf{s}$. Most important, $a_j(\mathbf{s} - \mathbf{d}_j)$ is referred to as the *propensity* function of the reaction R_j, that is the probability per unit time, of a change $\mathbf{d}_j$ occurring, given that we are in the state $\mathbf{s} - \mathbf{d}_j$.

With these definitions we can define the *Chemical Master Equation* (CME) (Gillespie, 1977):

$$\frac{dP_{\mathbf{s}}(t)}{dt} = \sum_{j=1}^{m} [a_j(\mathbf{s} - \mathbf{d}_j) P_{(\mathbf{s}-\mathbf{d}_j)}(t) - a_j(\mathbf{s}) P_{\mathbf{s}}(t)]. \tag{2.2}$$

This equation describes the probabilities of moving in or out of the state $\mathbf{s}$. For each state $\mathbf{s}$ we have then a differential-difference equation of this form. This equation has been derived using physical assumptions about the probability that the single molecules have to collide and therefore react. In particular, Gillespie (Gillespie, 1977) derived the parameter $c_j dt$, the average probability that a particular combination of R_j reactants molecules will react accordingly in the next infinitesimal time interval dt. The propensity function $a_j(\mathbf{s})$ is the product of c_j and $h_j(\mathbf{s})$, the number of distinct combinations of R_j reactant molecules. The term $c_j dt$ is called the *stochastic rate constant*.

It is interesting to remark that it has been proved there is a correspondence between c_j and the GMA rate constant k_j (Wolkenhauer et al., 2004):

$$c_j = \left(\frac{k_j}{(N_A V)^{K_j - 1}} \right) \cdot \prod_{z=1}^{L_j} (\kappa_{jz}!) \tag{2.3}$$

where N_A is the Avogadro number and V is the cell volume. This allows one to pass from one method to the other as soon as either c_j or k_j has been identified from experimental data.

2.2.5 Stochastic Simulation Algorithm

A major difficulty with the CME is that its analytical solution is usually intractable. For this reason, Gillespie (Gillespie, 1977) developed the *Stochastic Simulation Algorithm* (SSA), a Monte Carlo simulation of the CME. A single simulation represents one exact possible evolution of the system, while a set of thousands of these simulations can be used to identify a probability function that is an approximation of the CME.

This algorithm proceeds with a loop in which, at every iteration, two parameters are randomly taken from previously defined probability distributions: the time of the next reaction and which reaction will occur next. In order to compute these values, the joint probability that reaction R_j will be the next reaction and will occur in the infinitesimal time interval $[t, t+\delta t)$, given $(\#\mathbf{S} = \mathbf{s})$, is computed:

$$P(\tau, j | \mathbf{s}, t) = a_j(\mathbf{s}) e^{-a_0(\mathbf{s})\tau} \tag{2.4}$$

where $a_0(\mathbf{s}) = \sum_{j=1}^{m} a_j(\mathbf{s})$.

Starting from 2.4, the probabilities of the next reaction and the time of the next reaction can be obtained:

$$P(\tau | \mathbf{s}, t) = a_0(\mathbf{s}) e^{-a_0(\mathbf{s})\tau} \quad \tau \geq 0$$
$$P(j | \tau, \mathbf{s}, t) = \frac{a_j(\mathbf{s})}{a_0(\mathbf{s})} \quad j = 1, ..., M$$

From these distributions, random Monte Carlo samples can be taken using two uniform random numbers r_1 and r_2 from $[0, 1]$. Time delay τ is given by:

$$\tau = \frac{1}{a_0(\mathbf{s})} \ln\left(\frac{1}{r_1}\right) \tag{2.5}$$

The index j of the selected reaction is the smallest integer in $[1, m]$ such that

$$\sum_{j'=1}^{j} a_{j'}(\mathbf{s}) > r_2 a_0(\mathbf{s}) \tag{2.6}$$

Once these two values are computed, the system is updated adding the selected $\mathbf{d}_j$ to $\mathbf{s}$ and τ summed to t.

CME and SSA are very specific to the biochemical context. We now turn our attention to a more general type of stochastic model: continuous time Markov chains (CTMCs).

2.2.6 Deterministic and Stochastic Approaches

A *deterministic* approach to the modelling of biochemical reactions is characterised by producing the most likely behaviour of the system and expressing its output as the continuous concentration of the biochemical species. In contrast, a *stochastic* approach considers the likelihood of alternative behaviours and expresses its output as discrete quantities, such as number of molecules or levels of concentration.

Deterministic ODE based models are widely used in modelling biochemical interactions. They represent the most efficient approach, able to model hundreds of reactions at the same time. However, ODE does not account for randomness or stochasticity, which are key features of biochemical interactions (McAdams and Arkin, 1999). When experimental evidence for the modelled systems presents low variability or all the species in the systems are present in large quantities, we might consider ODEs to be the most suitable representation. However, even in systems which exhibit low variability this may be due to high resistance to noise and some behaviour may be lost with a deterministic approach.

An example of such behaviour is the circadian clock, a biochemical system that presents oscillations. It has been shown that only stochastic models allow

identification of which interactions are required for the oscillations to resist the randomness of the interactions (Barkai and Leibler, 2000). Another example can be found in the repressilator presented here, where experimental data highlights weak resistance to noise (Elowitz and Leibler, 2000).

In general, a stochastic approach offers a more accurate representation of the biological interactions under study. However, this type of approach presents the main drawback of being computationally expensive, thus limiting the size of the biochemical network that can be analysed.

2.2.7 Continuous Time Markov Chain

Definition 2.1 *Continuous time Markov chain.* A continuous time Markov chain (CTMC)(Ross, 1983) is a pair $(U, \rightarrow)$ where:

- U is a countable set of states;
- $\rightarrow$ is a set of transitions with $\rightarrow \subseteq (U \times \mathbb{R}_{>0} \times U)$. The real number associated with each transition is the rate for the exponential time delay for the transition to happen.

The set of transitions $\rightarrow$ can be interpreted as the *infinitesimal generator matrix* $Q = \{q_{ij}\}$, $i, j \in U$, which collects the rates of the transition from state i to j. The elements q_{ij} of the matrix Q are defined as follows:

$$\text{if } i \neq j \text{ then } q_{ij} = \begin{cases} r \text{ if } \exists r \in \mathbb{R}_{>0},\ (i, r, j) \in \rightarrow \\ 0 \text{ otherwise} \end{cases}$$

$$q_{ii} = -\sum_{j \neq i} q_{ij}$$

The probability $P_{ij}(t)$ to move from a state i to a state j within a time t is given by an exponential distribution with rate q_{ij}, i.e.

$$P_{ij}(t) = 1 - e^{-q_{ij}t}$$

In the presence of multiple transitions outgoing from a state i, a *race condition* is employed. A race condition implies that the transitions outgoing from a state i are in competition and that the faster transition will be triggered, determining the next state j. Once this happens a new race starts to determine the successive

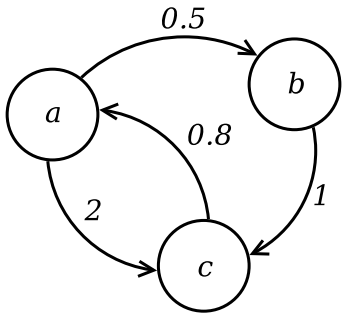


Figure 2.1: Example of a CTMC. Numbers on the transitions are exponential rates.

state. Because each transition is associated with an exponential time delay, the time delay to leave i will also be exponentially distributed with a rate equal to the sum of the rates of the outgoing transitions.

An Example of a CTMC is depicted in Figure 2.1. In this example, $U = \{a, b, c\}$ while the infinitesimal generator matrix is given by:

$$Q = \begin{pmatrix} -2.5 & 0.5 & 2 \\ 0 & -1 & 1 \\ 0.8 & 0 & -0.8 \end{pmatrix}$$

Sampling over a CTMC. Given the initial state $u \in U$ of a CTMC, a next state and a time delay can be sampled in analogy with SSA (Section 2.2.5).

Using $q_u = \sum_j q_{uj}$ and $rand_1$ and $rand_2$ uniform random numbers in $[0, 1]$, the time delay τ is given by:

$$\tau = \frac{1}{q_u} \ln\left(\frac{1}{rand_1}\right)$$

The index j of the selected reaction is the smallest integer in $[1, n]$ such that

$$\sum_{i=1}^{j} q_{ui} > rand_2 q_u$$

The sampling can be repeated until no transitions are possible, i.e. an *absorbent* state is reached, or until the sum of the time delays reaches a time threshold. A

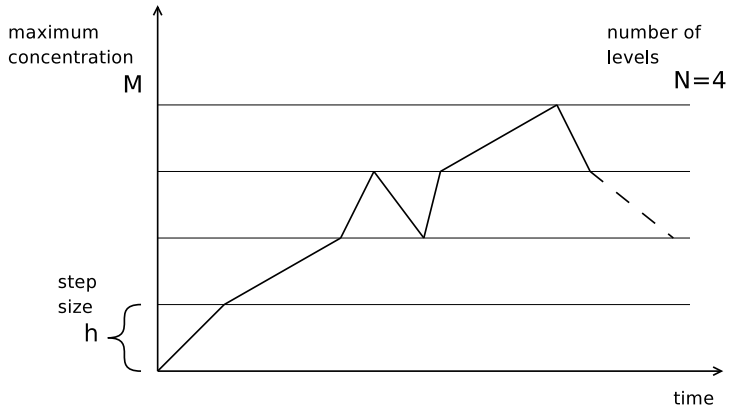


Figure 2.2: Illustration of the concept of levels of concentration.

sequence of samplings forms a *trace* of the form $u, \tau_1, u_1, \tau_2, u_2, \ldots, \tau_m, u_m$, also called a *Monte Carlo simulation* of the CTMC. A collection of simulations can be used to estimate the probability density function capturing the likelihood of the states of the CTMC in time.

In the next section we illustrate how CTMCs can be used to model biochemical reactions.

2.2.8 CTMC with Levels of Concentration

The CTMC with levels is a family of CTMCs that can be used to model biochemical species abstracting their concentration with discrete levels. The Markov chains belonging to this family have a common definition and differ only in the number of levels used for each species and the amount of concentration represented by one level. Models of biochemical interactions written in ODEs can be converted easily into CTMC with levels.

The original idea behind CTMC with levels was to represent signals in a cell, in terms of high and low concentration of species involved in signalling pathways (Calder et al., 2006b). It then became evident that an arbitrary number of levels could be used. Moreover, Kurtz's Theorem (Kurtz, 1971) provides a strong theory that links CTMCs with different number of levels with one another and that relates them to ODE models of the same modelled system.

In Figure 2.2 we illustrate the concept of levels of concentration. Initially, a preliminary investigation is necessary, in order to identify the maximum concentration $M \in \mathbb{R}^+$ for each species. Then a number $N \in \mathbb{N}$ is chosen to divide the

range of values into discrete levels $\{0, 1, ..., N\}$. The *step size* or granularity of the model is defined as $h = M/N$ and represents the amount of concentration represented by one level.

A state of a CTMC with levels of concentration is defined as the vector $\sigma = (\langle \mathbf{S}_1 \rangle, ..., \langle \mathbf{S}_n \rangle)$. The notation $\langle \mathbf{S}_i \rangle$ indicates the current level of concentration of the species $\mathbf{S}_i$, with $0 \leq \langle \mathbf{S}_i \rangle \leq N$. An index j is assigned to each biochemical reaction. The dynamics of the reactions are described by kinetic laws v_j, such as Mass Action or Michaelis-Menten. We consider the case in which each species has the same step size h.

The rates of the CTMC, used to pass from one state to another, are defined as follows. Let $d[\mathbf{S}_i]/dt = v_j(\mathbf{x}, \mathbf{d}_j)$ be a kinetic law, $\mathbf{S}_i$ one of the products, $\mathbf{x}$ the vector of concentrations of reactants and modifiers of reaction j and $\mathbf{d}_j = (d_{1,j}, ..., d_{n,j})$ the vector of the stoichiometric coefficients of reaction R_j. Consider the following linear approximation:

$$[\mathbf{S}_i]_{t'} \approx [\mathbf{S}_i]_t + v_j(\mathbf{x}_t, \mathbf{d}_j) \cdot (t' - t)$$

where $[\mathbf{S}_i]_t$ is the concentration of the species $\mathbf{S}_i$ at time t and $t' - t = \Delta t$ is the time difference between t' and t. We can now define the step size of the species $\mathbf{S}_i$ as $h = [\mathbf{S}_i]_{t'} - [\mathbf{S}_i]_t$ - the difference in concentration between two levels. With h fixed we can define:

$$\lambda_j = \frac{1}{\Delta t} = \frac{v_j(\mathbf{x}_t, \mathbf{d}_j)}{h}$$

where λ_j is defined as the rate, or parameter of an exponential distribution $g(t, j) = e^{-\lambda_j t}$, with mean $\mathrm{E}[g(t, j)] = 1/\lambda_j = \Delta t$. Since $\mathbf{x}_t = \sigma_t \cdot h$, λ_j is the rate of the reaction j and is a function of the current state σ_t. A reaction j brings the current state from σ_u to $\sigma_v = \sigma_u + d_j$.

An interesting theoretical result is derived from Kurtz's Theorem (Kurtz, 1971). It states that, in the limit of a decreasing step size in the CTMC, the most likely time evolution tends to the ODE simulation. For a more detailed explanation see (Ciocchetta et al., 2009).

2.2.9 Compartments

Compartments are spatial locations that abstract cells, organelles (e.g. mitochondria) or other entities that can contain molecular species and that are characterised by a volume.

Let $\mathbf{A}$ be a species inside a compartment C_a with volume V_a that can be transported to another compartment C_b with volume V_b, where it takes the new name $\mathbf{B}$. This can be represented by the following transport reaction R_t:

$$R_t : \mathbf{A} \longrightarrow \mathbf{B}$$

The ODEs for this model are:

$$v = k[\mathbf{A}] \qquad V_a \cdot \frac{d[\mathbf{A}]}{dt} = -v \qquad V_b \cdot \frac{d[\mathbf{B}]}{dt} = v \tag{2.7}$$

where v is the velocity of the transport in moles per time unit. The rates of the corresponding CTMC with levels can be derived as follows. From Equation (2.7), we can infer the rate λ of the exponential distribution of the time necessary to transport one level of concentration of $\mathbf{A}$ from C_a to C_b.

Consider the following difference equations:

$$\begin{aligned} & V_a \cdot \frac{\Delta[\mathbf{A}]}{\Delta t} = -v \qquad V_b \cdot \frac{\Delta[\mathbf{B}]}{\Delta t} = v \\ \Rightarrow \qquad & V_a \cdot \frac{\Delta\langle\mathbf{A}\rangle \cdot h_a}{\Delta t} = -v \qquad V_b \cdot \frac{\Delta\langle\mathbf{B}\rangle \cdot h_b}{\Delta t} = v \end{aligned}$$

where $\Delta\langle\mathbf{A}\rangle$ and $\Delta\langle\mathbf{B}\rangle$ are the changes in number of levels of $\mathbf{A}$ and $\mathbf{B}$ after the transport of one level of concentration from C_a to C_b. We assume these to be $\Delta\langle\mathbf{A}\rangle = -1$, i.e. $\mathbf{A}$ decreases one level and $\Delta\langle\mathbf{B}\rangle = 1$, i.e. $\mathbf{B}$ increases one level. We then obtain:

$$\lambda = \frac{1}{\Delta t} = \frac{v}{V_b \cdot h_b} = \frac{v}{V_a \cdot h_a} \tag{2.8}$$

It is worth noting that Equation (2.8) implies that $h_b = h_a \cdot V_a / V_b$.

2.2.10 Reaction-Diffusion Equations

So far we have assumed that biochemical reactions happen in a well-mixed environment. This implies that diffusion of molecules is so fast that whenever

biochemical reactions take place, molecules are immediately rearranged to a well-mixed solution. However, many processes in biology, such as pattern formation in development, can be modelled only if diffusion of molecules in space is represented explicitly.

Modelling diffusions of species $\mathbf{S}$ is defined at a macroscopic level by Fick's equation, a Partial Differential Equation (PDE) of the form:

$$\frac{\partial[\mathbf{S}]}{\partial t} = D_{\mathbf{S}} \nabla^2 [\mathbf{S}] \tag{2.9}$$

where $D_{\mathbf{S}}$ is the diffusion coefficient of species $\mathbf{S}$, $[\mathbf{S}]$ now represents the concentration density of $\mathbf{S}$ at a point in space and ∇^2 is the Laplacian operator, which can be interpreted in different ways, depending on the coordinate system. When diffusion is considered along with local biochemical interactions, the following reaction-diffusion equation (RDE) is employed (Berg, 1993; Jones and Sleeman, 1983):

$$\frac{\partial[\mathbf{S}]}{\partial t} = D_{\mathbf{S}} \nabla^2 [\mathbf{S}] \pm \text{React} \tag{2.10}$$

where React represents the velocities of other reactions involving species $\mathbf{S}$. In order to compute the concentration of $\mathbf{S}$ in a volume, $[\mathbf{S}]$ has to be integrated in that volume. An example of reaction-diffusion equation in one-dimensional coordinates is:

$$\frac{\partial[\mathbf{S}](t,x)}{\partial t} = D_{\mathbf{S}} \frac{\partial^2[\mathbf{S}](t,x)}{\partial x^2} - k_{deg}[\mathbf{S}](t,x) \tag{2.11}$$

where x indicates the position and t the time. In order to solve a RDE, initial conditions and boundary conditions need to be specified. In particular, boundary conditions are constraints to be applied at the edges of the spatial area considered. Usually they specify whether the boundaries *reflect* or *absorb* the concentration that reaches the edges. Models defined by reaction-diffusion equations can also be approximated by a CTMC with levels. In this case, space has to be divided into regions and the diffusion term of the equations is approximated by the mass action kinetic law. For details about this procedure see (Erban et al., 2007).

2.2.11 Modelling Cells and Tissues

Until now we have discussed modelling approaches used to represent biochemical reactions and movement of molecules. This is the most common level of detail for models of metabolic and signalling pathways. However, when we shift our interest to cells, tissues and organs, the complexity of the number of molecules and the interaction involved is such that molecular models become difficult if not prohibitive to analyse. Details are thus hidden in favour of explicit modelling of higher order structures. Moreover, at these scales, the molecular details of entities and interactions are often still unclear or unknown. In this scenario, *agent-based systems* (Macal and North, 2010) such as cellular automata are employed.

2.2.12 Cellular Automata

Cellular automata (CA) (Ermentrout and Keshet, 1993; Packard and Wolfram, 1985), are characterised by discrete states and discrete time steps. A set of simple rules is used to move from a state to another in a deterministic way. In detail, a state in CA is usually represented by a grid, where each cell can assume two or more states. At each discrete time step, rules are applied to update the state of each cell in the grid according to the state of its neighbouring cells.

CA have the advantage of being simple yet able to adapt to many different scenarios, such as models of diffusion, pattern formation and tumour growth. Most importantly, rules can be applied to the cells in parallel, ensuring fast simulations of large models that are often too complex for other approaches. Main criticisms to this approach are that CA might be too simple to provide significant insights to the biological phenomena modelled.

2.2.13 Multi-Scale Models

One of the most difficult problems in Biology is to understand how small simple parts like molecules work together to form complex organisms. The modelling of molecules and biochemical reactions in isolation is a relatively simple task, when compared to the modelling of more complex systems and processes such as the cardiovascular system or the morphogenesis of organs. A fundamental challenge comes from the fact that biological phenomena appear at different time scales, from milliseconds to years, and that levels of organisation of molecules

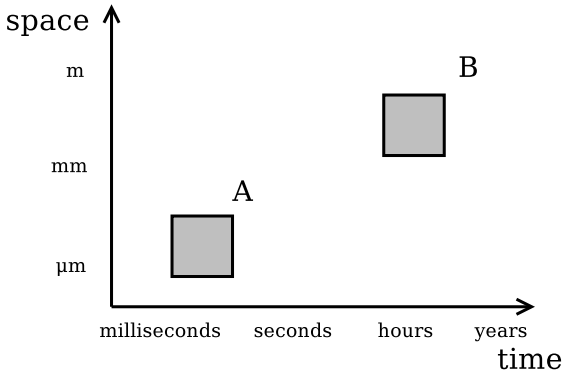


Figure 2.3: The scale separation map (Walker and Southgate, 2009). Depending on the biological system or the level of details chosen, models refer to specific time and spatial scales. Models A and B refer to two different time and spatial scales.

reach different spatial scales, from micrometres to metres. Usually, a modelling approach is able to model effectively only a specific time and spatial scale (Figure 2.3). As a consequence, approaches based on a combination of multiple modelling techniques have emerged (Dada and Mendes, 2011; Walker and Southgate, 2009). These approaches are usually tailored around a specific phenomenon. For example, models of tumour growth use agent-based models of tissue interactions, and ODEs models for biochemical reactions (Athale et al., 2005; Wang et al., 2005).

As a side note, multi-scale models are also of relevance in computing. For example, models of internet worms spread have been implemented using ODEs and process based approaches (Nicol, 2008).

2.3 Formal Modelling Methods

In this section we discuss formal approaches to the modelling of biological systems, with the main focus on process algebras. We begin introducing definitions of multi-sets, *labelled transition system* (LTS) and *rated* LTS.

2.3.1 Multi-Sets

In this section we give the definition of multi-sets that we will use throughout the thesis. We use $\{\!|$ and $|\!\}$ to delimit a multi-set. For example, $A = \{\!|5,\ 6,\ 6,\ 7,\ 7,\ 7|\!\}$ is a multi-set.

We define a multi-set M as the pair (M', m_M), where M' is a set containing the same elements of M with no repetitions and m_M is the associated multiplicity function, such that for all $x \in M'$, $m_M(x)$ is equal to the number of times x appears in M'. For all $x \in M'$, $m_M(x)$ is equal to zero if x does not appear in M'. For example, multi-set A is defined as the pair (A', m_A), where $A' \subset \mathbb{N}$, $A' = \{5,\ 6,\ 7\}$, and $m_A : \mathbb{N} \to \mathbb{N}$, $m_A(5) = 1$, $m_A(6) = 2$ and $m_A(7) = 3$.

Given two multi-sets A and B, defined as (A', m_A) and (B', m_B), the following operations are defined:

- multi-set union: $A \cup B = (A' \cup B', m_{A \cup B})$, where for all $x \in A' \cup B'$, $m_{A \cup B}(x) = \max(m_A(x), m_B(x))$;
- multi-set sum: $A \uplus B = (A' \cup B', m_{A \uplus B})$, where for all $x \in A' \cup B'$, $m_{A \uplus B}(x) = m_A(x) + m_B(x)$;
- multi-set intersection: $A \cap B = (A' \cap B', m_{A \cap B})$, where for all $x \in A' \cap B'$, $m_{A \cap B}(x) = \min(m_A(x), m_B(x))$;
- multi-set difference: $A \setminus B = (A', m_{A \setminus B})$, where for all $x \in A'$, $m_{A \setminus B}(x) = \min(0, m_A(x) - m_B(x))$.

Moreover we define:

- $A \subseteq B \Leftrightarrow$ for all $x \in A' \cup B'$, $m_A(x) \leq m_B(x)$;
- $|A| = \sum_{x \in A'} m_A(x)$.

2.3.2 Labelled Transition System

Definition 2.2 *Labelled transition system.* A labelled transition system (LTS) is a triple $(U, Lab, \to)$ where:

- U is a countable set of states;
- Lab is the set of labels;
- $\to$ is a multi-set of transitions with $\to \subseteq (U \times Lab \times U, m_\to)$ where $m_\to : (U \times Lab \times U) \to \mathbb{N}_{>0}$ indicates the multiplicity of each transition in $\to$. If $(a, x, b) \in \to$, we denote this also as the labelled transition $a \xrightarrow{x} b$.

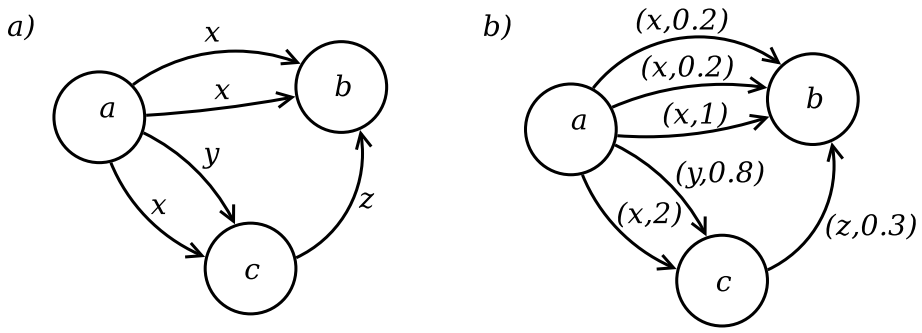


Figure 2.4: *a*) Example of a labelled transition system. *b*) Example of a rated labelled transition system.

An example of a LTS is depicted in Figure 2.4 *a*, where $U = \{a, b, c\}$, $Lab = \{x, y, z\}$ and $\rightarrow = \{\!|(a, x, b), (a, x, b), (a, x, c), (a, y, c), (c, z, b)|\!\}$.

Definition 2.3 *Rated labelled transition system.* A rated LTS is a triple $(U, Lab, \rightarrow)$ where:

- U is a countable set of states;
- Lab is the set of labels;
- $\rightarrow$ is a multi-set of transitions with $\rightarrow \subseteq (U \times Lab \times \mathbb{R}_{>0} \times U, m_{\rightarrow})$ where $m_{\rightarrow} : (U \times Lab \times \mathbb{R}_{>0} \times U) \rightarrow \mathbb{N}_{>0}$ indicates the multiplicity of each transition in $\rightarrow$. The real number associated with each transition is the rate for the exponential time delay for the transition to happen.

An example of a rated LTS is depicted in Figure 2.4 *b*, where $U = \{a, b, c\}$, $Lab = \{x, y, z\}$ and $\rightarrow = \{\!|(a, x, 0.2, b),\ (a, x, 0.2, b),\ (a, x, 1, b),\ (a, x, 2, c),\ (a, y, 0.8, c),\ (c, z, 0.3, b)|\!\}$.

Sampling over a rated LTS. Given a rated LTS $(U, Lab, \rightarrow)$ and initial state $u \in U$, we can sample the next state and a time delay in the same way as in the sampling over a CTMC (Section 2.2.7). In particular, a CTMC can be obtained from a rated LTS. The infinitesimal generator matrix $Q = \{q_{ij}\}$, $i, j \in U$, can be obtained as follows:

$$\text{if } i \neq j \text{ then } q_{ij} = \begin{cases} r \text{ if } \exists x,\ (i, a, x, j) \in \rightarrow \text{ and } r = \sum_{k \in K(i,j)} k \\ 0 \text{ otherwise} \end{cases}$$

$$q_{ii} = -\sum_{j \neq i} q_{ij}$$

where $K(i,j) = \{\!|k \mid (i,a,k,j) \in \!\rightarrow |\!\}$.

2.3.3 An Introduction to Process Algebra

Fundamental concepts. A process algebra is characterised by a *syntax* and by one or more *semantics*. While the former defines how a process algebra model is written, the latter defines how the behaviour of a model is determined from its syntactic definition. The fundamental elements in a process algebra are autonomous agents called *processes*. Each process is characterised by its behaviour, expressed in terms of *actions* it can perform. For example, if a process P performs a sequence of three a actions, we can denote it as:

$$P \triangleq a.a.a.nil$$

where "." is the sequential operator and *nil* is defined as the deadlock process, i.e. the process that cannot perform any action. A *labelled transition* is usually employed to show that process can perform an action and become another process. For example, P can perform action a and become process P', defined as $P' \triangleq a.a.nil$. This is denoted as:

$$P \xrightarrow{a} P'$$

Process P' is called a *one-step derivative* of P, while if a process can be obtained after any number of transitions from P, this is called simply a *derivative* of P. The set of derivatives of a processes and all the labelled transitions from such derivatives form a *derivation graph*, which is an LTS where the set of states U is the set of all derivatives and the set of labels *Lab* is the set of all actions.

It could be the case that process P can choose non deterministically between multiple actions available. This is denoted using the *choice* operator "+". For example:

$$Q \triangleq a.nil + b.nil + c.d.nil$$

Here Q can produce the following three labelled transitions:

$$Q \xrightarrow{a} nil \qquad Q \xrightarrow{b} nil \qquad Q \xrightarrow{c} d.nil$$

Most importantly, processes can synchronise on actions. Synchronisation can be *binary* between two actions with complementary names, in the style of calculus of communicating systems (CCS) (Milner, 1989), or *multi-way* between any number of actions sharing the same name, in the style of communicating sequential processes (CSP) (Hoare, 1985) and later of performance evaluation process algebra (PEPA) (Hillston, 1996). As we anticipated in the introduction to this thesis, we follow the latter approach. Multi-way synchronisation is possible using the *cooperation* operator $\underset{\mathcal{L}}{\bowtie}$. The set of actions $\mathcal{L}$, or cooperation set, indicates which actions are used for synchronisation. For example, given the processes:

$$R \triangleq a.nil + b.nil \qquad S \triangleq a.nil + b.nil + c.nil$$

and the overall model defined as $R \underset{\{a,c\}}{\bowtie} S$, we have that only the following transitions are possible:

$$R \underset{\{a,c\}}{\bowtie} S \xrightarrow{a} nil \underset{\{a,c\}}{\bowtie} nil \qquad R \underset{\{a,c\}}{\bowtie} S \xrightarrow{b} nil \underset{\{a,c\}}{\bowtie} S \qquad R \underset{\{a,c\}}{\bowtie} S \xrightarrow{b} R \underset{\{a,c\}}{\bowtie} nil$$

Because action a is in the cooperation set, R and S can synchronise on a, but cannot perform a individually. On the contrary, b is not in the cooperation set, so R and S cannot synchronise on b, though they can perform b individually. Finally, c cannot be performed by S, because it is present in the cooperation set, which would require that also R had the possibility of performing c.

Another key feature of process algebra is the possibility for actions to become *hidden*. This is usually expressed by replacing the name of an action with the *hidden action type* τ. This substitution may happen in an *implicit* way, as in CCS, or in an *explicit* way, as in CSP. In CCS, as a result of a binary synchronisation, the name of the two complementary actions that synchronise is replaced by τ. As no other actions will synchronise, there is no need for the rest of the system to know what specific action took place, information that is therefore considered internal to the participants to the synchronisation. In contrast, in CSP there is no way to know how many processes will synchronise, because of the multi-way synchronisation. For this reason it is the responsibility of the modeller to place *hiding* ($\backslash$) operators appropriately in the system. For example, with $R \underset{\{a,c\}}{\bowtie} S \backslash \{b\}$ we impose that if $R \underset{\{a,c\}}{\bowtie} S$ can perform action b, that action will be replaced with τ upon application of $\backslash\{b\}$. This results in the following labelled transitions:

$$(R \underset{\{a,c\}}{\bowtie} S) \setminus \{b\} \xrightarrow{a} (nil \underset{\{a,c\}}{\bowtie} nil) \setminus \{b\} \qquad (R \underset{\{a,c\}}{\bowtie} S) \setminus \{b\} \xrightarrow{\tau} (nil \underset{\{a,c\}}{\bowtie} S) \setminus \{b\}$$
$$(R \underset{\{a,c\}}{\bowtie} S) \setminus \{b\} \xrightarrow{\tau} (R \underset{\{a,c\}}{\bowtie} nil) \setminus \{b\}$$

Action Priorities. Action priorities have been introduced in the process algebra theory to model interrupt mechanisms in computer systems (Baeten et al., 1986; Cleaveland and Hennessy, 1990). In this extension, $p{:}a$ indicates that action a is performed with priority p, with $p \in (\mathbb{N} \setminus \{0\})$. Actions with higher priority have precedence and block actions with lower priority. Consider the following process:

$$Q \triangleq 1{:}a.nil + 2{:}b.nil + 3{:}c.1{:}d.nil$$

The only enabled labelled transition is $Q \xrightarrow{c} 1{:}d.nil$, because action c has the highest priority if compared with the other available actions a and b. Moreover, consider the following processes:

$$R \triangleq 1{:}a.nil + 2{:}b.nil \qquad S \triangleq 1{:}a.nil + 2{:}b.nil + 3{:}c.nil$$

In this case, the only enabled transitions are:

$$R \underset{\{a,c\}}{\bowtie} S \xrightarrow{b} nil \underset{\{a,c\}}{\bowtie} S \qquad R \underset{\{a,c\}}{\bowtie} S \xrightarrow{b} R \underset{\{a,c\}}{\bowtie} nil$$

Because although the action with highest priority is c, c is not available, i.e. it cannot be performed by process $R \underset{\{a,c\}}{\bowtie} S$. This is because c is in the cooperation set of $\underset{\{a,c\}}{\bowtie}$ and is available in S, but not in R.

Relations. Finally, *relations* are usually defined between processes, to express to which extent the behaviour of two processes can be considered the same. In particular, fundamental to every process algebra theory are the notions of *equivalences*, such as *isomorphism* or *bisimulation* (Milner, 1989), and whether such equivalences are also *congruences* or not. In general, if an equivalence relation is a congruence it ensures that the substitution of a process within a system with an equivalent process will produce a system which is identical to the original one, at least with respect to the behaviour preserved by the chosen equivalence.

2.3.4 Process Algebras for Biology

Modelling biological systems with existing formalisms, either algebras, calculi or languages, requires an abstraction, i.e. a matching between biological entities and the syntactic components of the chosen formalism.

In process algebra, this was first introduced with the "molecule-as-computation" abstraction (Regev et al., 2001), using *π-calculus* (Milner, 1999) where each

molecule in a system is represented by one or more concurrent processes, abstracting the behaviour and/or interactions it might have. Two molecules can react when their corresponding processes share a complementary channel, abstracting complementary binding sites. An example is the following process definition of a molecule **M**:

$$M \triangleq (\nu m)x!m.(m!a + m?a) + x?p.(p!a + p?a)$$

where $x!m$ represents the action of sending the name m into channel x and $x?p$ is the action of receiving from channel x a name that will replace p and all its occurrences in the local sequence of actions. Symbols '.', '+' and '|' are respectively the sequential operator, the choice operator and the parallel operator. Synchronisation can happen only in a binary fashion, when a *send* and a *receive* action that are divided by a parallel operator are executed together. Symbol νm is the scope restriction of the name m, and it is often used to delimit the processes that define a molecule or to restrict interactions, implementing compartmentalisation.

On the one hand, π-calculus is mathematically well understood and brings with it many associated tools. On the other hand, it may be a too low-level language. The language *beta binders* (Priami and Quaglia, 2005) tries to improve this aspect by enriching the syntax inserting the processes that abstract the behaviour of a biological entity in a box. Boxes are allowed to communicate internally, as in π-calculus, and externally, through special channels, or interfaces. A set of types is assigned to each channel that is used for external communications. Two boxes can then interact if they share at least one type, leading to more flexibility and higher non-determinism. Finally, boxes can fuse or divide using two types of function, *join* and *split*, that are essentially rewriting rules. This is an example of the parallel instantiation of two beta binders bio-processes:

$$\beta(x : \{a, b\})[!xz.P_1|Q_1] \parallel \beta(y : \{a, c\})[?yu.P_2|Q_2]$$

where the boxes containing an extended version of π-calculus processes are delimited by the brackets $[\cdot]$ and the interfaces are declared within the $\beta(\cdot)$. In this case x is an external channel with types a and b. The two boxes can interact via channels x and y, because they share a common type a.

The languages π-calculus and beta binders present similar drawbacks. They are highly expressive languages that allow many ways to model the same system, requiring experts to perform the modelling. Moreover, the level of detail that is considered might be too high for the usual need of biologists. This is because often not all the interactions are well understood and, considering quantitative analysis, it is extremely difficult to obtain reliable stochastic rates for all the events or a sufficient amount of measurements for all the states an entity can assume.

A different approach is taken by *PEPA* (Hillston, 1996). The syntax of this language is much simpler, without the notion of passing of information through communication. An important characteristic is the synchronisation not only between two processes that present complementary channels, but between all processes that share the same available action name, allowing the modelling of reactions that involve an arbitrary number of molecular species. Because of this, the PEPA approach has the flexibility of abstracting more than one biological interaction with a single action.

Although modelling each molecule as a process is possible, a population approach has been preferred in current PEPA applications. This means that a process represents the number of molecules or the level of concentration of a species. After an event is observed, these processes change to new processes expressing different quantities. This approach is denoted "species-as-process" abstraction. This is an example:

$$A_H \triangleq (\alpha_1, r_1).A_L \qquad A_L \triangleq (\alpha_2, r_2).A_H$$

$$B_H \triangleq (\alpha_2, r_2).B_L \qquad B_L \triangleq (\alpha_1, r_1).B_H$$

$$A_H \underset{\{\alpha_1,\alpha_2\}}{\bowtie} B_L \xrightarrow{\alpha_1} A_L \underset{\{\alpha_1,\alpha_2\}}{\bowtie} B_H$$

Here two species **A** and **B** are modelled using processes that represent their amount in the system, either high or low. α_1 and α_2 are the actions on which the processes can synchronise, while r_1 and r_2 are the rates at which the actions occur. The labelled transition shows what happens if α_1 takes place from an initial system $A_H \underset{\alpha_1,\alpha_2}{\bowtie} B_L$.

Bio-PEPA (Ciocchetta and Hillston, 2009) is an extension of PEPA specifically designed to model biochemical interactions with a "species-as-process" ab-

straction. There are several improvements that are introduced in Bio-PEPA with respect to PEPA. First of all, it introduces syntax elements to better describe the nature of the interactions. In addition there is the possibility to define functions that express the reaction velocities, based on biochemical kinetics, that can be used to generate both ODEs and CTMCs. Finally, Bio-PEPA supports the description of a species in terms of its level of concentration. For example consider the following Bio-PEPA model of the interactions between two molecules **A** and **B**. The first line gives the process definitions, while the second line a transition from the initial system.

$$A \triangleq (prod, 1)\uparrow A + (mm, 1)\downarrow A \qquad B \triangleq (mm, 1)\uparrow B + (deg, 1)\downarrow B$$

$$A(10) \underset{\{mm\}}{\bowtie} B(0) \xrightarrow{mm} A(9) \underset{\{mm\}}{\bowtie} B(1)$$

As it can be seen in the initial state $A(10) \underset{\{mm\}}{\bowtie} B(0)$, Bio-PEPA introduces parametric processes to keep track of the current amount of each species in terms of levels of concentration (here 10 for **A** and 0 for **B**). Process definitions become species definitions, a compact list of possible actions, such as *prod*, with associated stoichiometry and a symbol that indicates the effect of an action on the population of a species, such as $\uparrow$ for increase. A functional rate, not shown, is associated with each action.

A drawback of PEPA and Bio-PEPA with respect to π-calculus is the need to define each species and complex that may form (Calder and Hillston, 2009). In π-calculus and related calculi the combinatorial problem of complex formation or internal molecular modifications is addressed by the calculus and not by the modeller. As a consequence, in PEPA the state of the system is given by a process for each species, indicating its current amount while in π-calculus the state of the system is given by one or more processes for each molecule, with possibly many copies of the same processes indicating the presence of multiple copies of the same species.

Several process algebras have been defined or extended to integrate a notion of locality of the biochemical interactions. The P-systems (Păun and Rozenberg, 2002) and Brane Calculi (Cardelli, 2005) focus on computing with and on dynamic membranes. Dynamic compartments are exploited by Bioambients (Regev et al., 2004), a biological version of Mobile Ambient (Cardelli and Gordon, 2000), a

language developed for mobile computation. Static compartments are used in Bio-PEPA (Ciocchetta and Guerriero, 2009).

In general, every language presents the possibility of performing different analyses starting from the same description, automatically implementing ODEs, CTMCs or a version of SSA. Most process algebras have tools for simulations and analysis of models. Some of them are Spim (the stochastic Pi Machine) (Phillips and Cardelli, 2004), the Beta Workbench (Dematté et al., 2008), the PEPA Eclipse Plug-in (Tribastone et al., 2009) and the Bio-PEPA Workbench (Duguid, 2009).

Additionally, formal models of biological systems can be analysed using model checking, testing properties written in *continuous stochastic logic* (Heath et al., 2008).

2.3.5 Related formalisms

Formal definitions of biological systems that aim to facilitate maintenance and sharing of models are descriptive languages, e.g. SBML (Hucka et al., 2003), and graphical notations, e.g. Kitano Map (Kitano, 2003).

Formal modelling of biochemical interactions that are closely related to process algebras are those based on *rewriting rules*, such as κ-calculus (Danos et al., 2007), BioNetGen (Blinov et al., 2004), and Pathway Logic (Talcott, 2008). In these cases, the definition of a molecule is given only by its name and the state of its binding sites, leaving the definition of the possible interactions and modifications to the rewriting rules. This leaves more flexibility to the possible evolutions of a system, although it introduces the problem of writing unambiguous rules. A key feature is, like in π-calculus, that there is no need to state every molecule and complex that may form due to interactions or modifications. Two examples of formalisms based on rewriting rules that support compartmentalisation are bioκ-calculus (Laneve and Tarissan, 2007) and stochastic bigraphs (Krivine et al., 2008).

Finally, yet another approach involves the definition of high level languages, presented as umbrella descriptions that can potentially be converted to any of the mentioned formalisms. Two examples are BIOCHAM (Calzone et al., 2006) and LBS (Pedersen and Plotkin, 2008).

2.3.6 Strong and Markovian Bisimulations

In this section we give more details about strong bisimulation and Markovian bisimulation in process algebra.

Strong bisimulation. A binary relation $\mathcal{R}$ on $\mathbb{P}$ (the set of processes) is a *strong bisimulation* if whenever $P\ \mathcal{R}\ Q$:

- for all P' with $P \xrightarrow{a} P'$, there is Q' such that $Q \xrightarrow{a} Q'$ and $P'\ \mathcal{R}\ Q'$;
- for all Q' with $Q \xrightarrow{a} Q'$, there is P' such that $P \xrightarrow{a} P'$ and $P'\ \mathcal{R}\ Q'$.

Strong bisimilarity, written $\sim$, is the union of all strong bisimulations:

$$\sim = \bigcup\{\mathcal{R} \mid \mathcal{R} \text{ is a strong bisimulation}\}$$

Thus $P \sim Q$ holds if there is a strong bisimulation $\mathcal{R}$ with $P\ \mathcal{R}\ Q$. It can be shown that $\sim$ is an equivalence relation (Milner, 1989).

Markovian bisimulation. Consider labelled transitions where, along with the action performed by a process, a rate for the action is present, i.e. transitions of the form $P \xrightarrow{a,r} P'$, where $r \in \mathbb{R}^{>0}$. Usually, the semantics of an algebra, e.g. PEPA (Hillston, 1996), determines the rate. Here we consider rates as parameters of exponential distributions of the time necessary for actions to be performed.

Consider now the following two processes:

$$P \triangleq (a,1).P' + (a,1).P' \qquad Q \triangleq (a,2).Q'$$

The two processes are not strong bisimilar, because $P \xrightarrow{a,1} P'$ while $Q \xrightarrow{a,2} P'$. However, they are in some sense equivalent, because the sum of the rates from P to P' via action name a is 2, as from Q to P'. In other words, the probability of moving from P to P' in a certain time via a is identical to the the probability of moving from Q to P' in the same time via a. This equality is captured by Markovian bisimilarity.

An equivalence relation $\mathcal{R}$ on processes is a *Markovian bisimulation* if

$$P\mathcal{R}Q \iff \forall a \in Act\ \forall \mathcal{C} \in \mathbb{P}/\mathcal{R},\ \nu(P,a,\mathcal{C}) = \nu(Q,a,\mathcal{C})$$

where Act is the set of all action names, $\mathbb{P}$ is the set of all processes, $\mathbb{P}/\mathcal{R}$ is the set of all equivalence classes of the equivalence relation $\mathcal{R}$ and $\nu(P \xrightarrow{a} \mathcal{C})$ is the sum of all the rates from P to all processes in $\mathcal{C}$ via action name a.

Markovian bisimilarity, written $\sim_m$, is the union of all Markovian bisimulations:

$$\sim_m = \bigcup\{\mathcal{R} \mid \mathcal{R} \text{ is a Markovian bisimulation}\}$$

Thus $P \sim_m Q$ holds if there is a Markovian bisimulation $\mathcal{R}$ with $P\ \mathcal{R}\ Q$. It can be shown that $\sim_m$ is an equivalence relation (Hillston, 1996).

Markovian bisimulation is based on *probabilistic bisimulation* (Larsen and Skou, 1991) and the *lumpability* property of CTMCs (Kemeny and Snell, 1960). A CTMC is lumpable if its states can be partitioned and aggregated preserving the Markov property.

2.4 Summary

In this chapter we have introduced and discussed the background material of this thesis. First, we gave an overview of related biological concepts (Section 2.1). Second, we illustrated some of the most popular mathematical approaches to the modelling of biological systems, with a main focus on the modelling of molecular interactions (Section 2.2). Finally, we discussed formal approaches, with main focus on the process algebra theory and its application to the modelling of biological systems (Section 2.3). In the next chapter we investigate the use of a simple process algebra with multi-way synchronisation to model biochemical interactions and tissue growth.

Chapter 3

Single-Scale Modelling with Process Algebra with Multi-way Synchronisation

In this chapter we show how biological systems can be modelled with a process algebra with multi-way synchronisation, focussing on a single scale. First, we discuss how to represent biological entities and events with processes and actions, using a simple process algebra with multi-way synchronisation (Section 3.1). Second, we augment the algebra with functional rates, obtaining a stochastic simple process algebra (Section 3.2.1). With this augmented algebra, quantitative models can be constructed. Finally, we show how processes can be parametrised to reduce the model definition (Section 3.3). We give example models of biochemical reactions and tissue growth (Sections 3.1.1, 3.1.2, 3.2.5, 3.3.2).

3.1 Simple Process Algebra

A minimal syntax of a process algebra with multi-way synchronisation (SPA) is:

$$P ::= nil \mid a.P \mid P + P \mid P \underset{\mathcal{L}}{\bowtie} P \mid A$$

where:

- P is a process, $P \in \mathbb{P}$, with $\mathbb{P}$ the set of processes;

Prefix

$$\frac{}{a.P \xrightarrow{a} P}$$

Choice Left

$$\frac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'}$$

Choice Right

$$\frac{Q \xrightarrow{a} Q'}{P + Q \xrightarrow{a} Q'}$$

Coop Left

$$\frac{P \xrightarrow{a} P'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{a} P' \underset{\mathcal{L}}{\bowtie} Q} \text{ if } a \notin \mathcal{L}$$

Coop Right

$$\frac{Q \xrightarrow{a} Q'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{a} P \underset{\mathcal{L}}{\bowtie} Q'} \text{ if } a \notin \mathcal{L}$$

Synchronisation

$$\frac{P \xrightarrow{a} P' \quad Q \xrightarrow{a} Q'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{a} P' \underset{\mathcal{L}}{\bowtie} Q'} \text{ if } a \in \mathcal{L}$$

Agent

$$\frac{P \xrightarrow{a} P'}{A \xrightarrow{a} P'} \text{ if } A \triangleq P$$

Figure 3.1: Semantics of a simple process algebra with multi-way synchronisation.

- *nil* is the deadlock process;
- a is an action, $a \in \mathit{Actions}$, with $\mathit{Actions}$ the set of actions;
- $a.P$ expresses the fact that action a has to be performed in order to change process $a.P$ into the new process P;
- $P + P$ expresses the non deterministic choice between two processes. Once one is chosen, the other is discarded;
- $P \underset{\mathcal{L}}{\bowtie} P$ expresses the cooperation between two independent processes via the cooperation set $\mathcal{L}$, with $\mathcal{L} \subseteq \mathit{Actions}$;
- A is used to define processes recursively, via the agent definition $A \triangleq P$. This implies that process P can be substituted with the agent name A.

The semantics for this syntax, given in operational semantics (Plotkin, 1981), is shown in Figure 3.1. The semantics produces a *labelled transition system* (see Section 2.3.2).

Using SPA we can describe behaviour at a scale or, in other words, the behaviour of biological entities observed at a certain level of detail. For example,

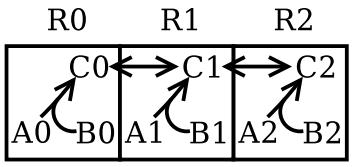


Figure 3.2: Biochemical reactions and transport between three regions in space.

we can model biochemical reactions, transport of biochemical species or tissue growth. We illustrate these examples in the next sections.

3.1.1 Simple Process Algebra and Biochemistry

The concentration of biochemical species can be abstracted using processes that represent levels of concentration (Calder et al., 2006b). If we use one process for each concentration level, a maximum number of levels N has to be specified. For example, given a biochemical species **A**, and $N = 10$, we can define process A_0 to represent zero concentration and process A_{10} to represent that the concentration of **A** has reached its maximum. Actions are used to abstract biochemical reactions, leading to jumps between concentration levels. If compartments or spatial regions are considered, transport of biochemical species can be modelled analogously.

As an example, consider three adjacent regions R0, R1 and R2. In each region, three types of biochemical species may be present: **A**, **B** and **C**. Molecules of **A** can react with molecules of **B**, yielding molecules of **C**. Concentration of **C** can also be transported between adjacent regions. In order to distinguish species in different regions, we write **A**1, for example, to indicate species **A** in region R1. This is illustrated in Figure 3.2. Finally, we use process $A1_2$ to indicate that in region 1, species **A** has concentration level 2.

The biochemical reactions are as follows:

$$R_{a0} : \mathbf{A}0 + \mathbf{B}0 \rightarrow \mathbf{C}0 \qquad R_{a1} : \mathbf{A}1 + \mathbf{B}1 \rightarrow \mathbf{C}1$$
$$R_{a2} : \mathbf{A}2 + \mathbf{B}2 \rightarrow \mathbf{C}2 \qquad R_{t01} : \mathbf{C}0 \rightarrow \mathbf{C}1$$
$$R_{t10} : \mathbf{C}1 \rightarrow \mathbf{C}0 \qquad R_{t12} : \mathbf{C}1 \rightarrow \mathbf{C}2$$
$$R_{t21} : \mathbf{C}2 \rightarrow \mathbf{C}1$$

We can model this system using the following processes:

$$
\begin{array}{lll}
A0_0 \triangleq nil & B0_0 \triangleq nil & C0_0 \triangleq a0.C0_1 + t10.C0_1 \\
A0_1 \triangleq a0.A0_0 & B0_1 \triangleq a0.B0_0 & C0_1 \triangleq a0.C0_2 + t10.C0_2 + t01.C0_0 \\
A0_2 \triangleq a0.A0_1 & B0_2 \triangleq a0.B0_1 & C0_2 \triangleq t01.C0_1 \\
A1_0 \triangleq nil & B1_0 \triangleq nil & C1_0 \triangleq a1.C1_1 + t21.C1_1 + t01.C1_1 \\
A1_1 \triangleq a1.A1_0 & B1_1 \triangleq a1.B1_0 & C1_1 \triangleq a1.C1_2 + t21.C1_2 + t12.C1_0 \\
 & & \quad + t01.C1_2 + t10.C1_0 \\
A1_2 \triangleq a1.A1_1 & B1_2 \triangleq a1.B1_1 & C1_2 \triangleq t12.C1_1 + t10.C1_1 \\
A2_0 \triangleq nil & B2_0 \triangleq nil & C2_0 \triangleq a2.C2_1 + t12.C2_1 \\
A2_1 \triangleq a2.A2_0 & B2_1 \triangleq a2.B2_0 & C2_1 \triangleq a2.C2_2 + t12.C2_2 + t21.C2_0 \\
A2_2 \triangleq a2.A2_1 & B2_2 \triangleq a2.B2_1 & C2_2 \triangleq t21.C2_1
\end{array}
$$

The initial state of the model is defined by the following process:

$$(A0_2 \underset{\mathcal{L}}{\bowtie} (B0_2 \underset{\mathcal{L}}{\bowtie} C0_0)) \underset{\mathcal{K}}{\bowtie} (A1_2 \underset{\mathcal{L}'}{\bowtie} (B1_2 \underset{\mathcal{L}'}{\bowtie} C1_0)) \underset{\mathcal{K}'}{\bowtie} (A2_2 \underset{\mathcal{L}''}{\bowtie} (B2_2 \underset{\mathcal{L}''}{\bowtie} C2_0))$$

where $\mathcal{L} = \{a0\}$, $\mathcal{L}' = \{a1\}$, $\mathcal{L}'' = \{a2\}$, $\mathcal{K} = \{t01, t10\}$ and $\mathcal{K}' = \{t12, t21\}$.
An example of a valid transition is:

$$(A0_2 \underset{\mathcal{L}}{\bowtie} (B0_2 \underset{\mathcal{L}}{\bowtie} C0_0)) \underset{\mathcal{K}}{\bowtie} (A1_2 \underset{\mathcal{L}'}{\bowtie} (B1_2 \underset{\mathcal{L}'}{\bowtie} C1_0)) \underset{\mathcal{K}'}{\bowtie} (A2_0 \underset{\mathcal{L}''}{\bowtie} (B2_0 \underset{\mathcal{L}''}{\bowtie} C2_2))$$
$$\xrightarrow{a1}$$
$$(A0_2 \underset{\mathcal{L}}{\bowtie} (B0_2 \underset{\mathcal{L}}{\bowtie} C0_0)) \underset{\mathcal{K}}{\bowtie} (A1_1 \underset{\mathcal{L}'}{\bowtie} (B1_1 \underset{\mathcal{L}'}{\bowtie} C1_1)) \underset{\mathcal{K}'}{\bowtie} (A2_0 \underset{\mathcal{L}''}{\bowtie} (B2_0 \underset{\mathcal{L}''}{\bowtie} C2_2))$$

where, following the execution of action $a1$, processes $A1_2$, $B1_2$ and $C1_0$ change into processes $A1_1$, $B1_1$ and $C1_1$ indicating that reaction R_{a1} took place in region R1 and that concentration levels have been updated accordingly. Another valid transition is:

$$(A0_2 \underset{\mathcal{L}}{\bowtie} (B0_2 \underset{\mathcal{L}}{\bowtie} C0_0)) \underset{\mathcal{K}}{\bowtie} (A1_2 \underset{\mathcal{L}'}{\bowtie} (B1_2 \underset{\mathcal{L}'}{\bowtie} C1_0)) \underset{\mathcal{K}'}{\bowtie} (A2_0 \underset{\mathcal{L}''}{\bowtie} (B2_0 \underset{\mathcal{L}''}{\bowtie} C2_2))$$
$$\xrightarrow{t21}$$
$$(A0_2 \underset{\mathcal{L}}{\bowtie} (B0_2 \underset{\mathcal{L}}{\bowtie} C0_0)) \underset{\mathcal{K}}{\bowtie} (A1_2 \underset{\mathcal{L}'}{\bowtie} (B1_2 \underset{\mathcal{L}'}{\bowtie} C1_1)) \underset{\mathcal{K}'}{\bowtie} (A2_0 \underset{\mathcal{L}''}{\bowtie} (B2_0 \underset{\mathcal{L}''}{\bowtie} C2_1))$$

where, following the execution of action $t21$, processes $C2_2$ and $C1_0$ change into processes $C2_1$ and $C1_1$, indicating that transport of concentration of **C** took place from region R2 to region R1.

A disadvantage of SPA is that every region requires the definition of its own processes, yielding long and repetitive process algebra descriptions. This will be improved with the introduction of parametric processes and actions (Section 3.3).

To give an idea of the different style of modelling of the π-calculus with respect to SPA, we reproduce here an example of a biochemical reaction $R_a : \mathbf{A} + \mathbf{B} \rightarrow \mathbf{C}$ from (Regev, 2002).

In π-calculus all molecules present in the system are modelled. In the case of reaction R_a we need definitions of processes representing molecules **A** and **B** in both *bound* and *unbound* states. Definitions are as follows:

$$\begin{aligned} A &\triangleq (\nu x)(bind!x.BoundA(x)) \\ BoundA(x) &\triangleq x!a.A \\ B &\triangleq bind?y.BoundB(y) \\ BoundB(y) &\triangleq y?b.B \end{aligned}$$

where $(\nu x)(P)$ means that the channel name x is private to process P, $bind!x$ is an *send* action, expressing the sending of action name x through channel *bind*, and $bind?y$ is a *receive* action, expressing the receiving of name y through channel *bind*. Processes $BoundA(x)$ and $BoundB(y)$ are parametric processes.

The initial state of the system is given by the cooperation between A and B processes, for example:

$$A|B|A|B|A|B$$

where there are three A processes representing three **A** molecules and three B processes representing three **B** molecules. At the execution of action $bind!x$ by a process A, private channel x is shared with a process B, by synchronisation with action $bind?y$. Selection of exactly one process A and one process B for the synchronisation is possible because synchronisation in π-calculus is binary and can happen only between a send and a receive sharing the same channel name (*bind*). This results in the new state:

$$(\nu x)(BoundA(x)|BoundB(x))|A|B|A|B$$

The process $(\nu x)(BoundA(x)|BoundB(x))$ indicates that a molecule **A** and a molecule **B** are bound. The fact that processes $BoundA(x)$ and $BoundB(x)$ share a private channel x indicates that the corresponding molecules are bound with one another. Thus $(\nu x)(BoundA(x)|BoundB(x))$ represents a product molecule **C**.

The advantage of using π-calculus with respect to SPA is that greater level of detail can be reached, with the possibility of representing and observing the behaviour of single molecules. Multiple copies of the same process are included in the system to indicate the presence of a certain quantity of a biochemical species. Because of the sharing of private channels, it is possible to identify processes that have interacted or can interact, representing complex formation and compartmentalisation.

The disadvantage of using π-calculus with respect to SPA is that it is much more complex to write and understand, while the binary as opposed to multi-way synchronisation may force the modeller to write models in a greater level of details that is actually needed, such as in the case of biochemical reactions with many reactants and many products.

Now we turn our attention to how spatial regions can be modelled in SPA.

3.1.2 Simple Process Algebra and Tissue Growth

In this section we show how tissue growth can be modelled in SPA. In particular we show that, because of multi-way synchronisation, the most suitable way to represent tissue growth is to define a finite area of space organised into regions, such as a grid, and model explicitly the available finite empty regions along with the tissue. We propose a model based on *explicit* modelling of empty space, followed by a discussion of issues that occur in models with *implicit* modelling of empty space.

Explicit Modelling of Empty Space. In this setting, processes represent either empty regions of space or regions containing tissue. Tissue can become empty space through tissue death or can interact with surrounding empty space, converting it into new adjacent tissue via tissue replication. When tissue is surrounded by tissue, replication is inhibited. Moreover, we use a process for each region in space we want to consider. This defines an area outside which growth

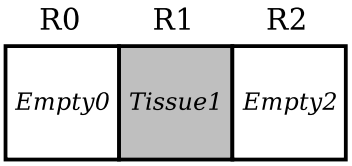


Figure 3.3: Explicit modelling of empty space. Every region in space has its associated process, even if it is empty space.

is not permitted and guarantees that the model will generate a finite state space. For simplicity, we consider three regions, depicted in Figure 3.3.

Tissue can either die (*apoptosis*) and become empty space or perform duplication (*mitosis*) expanding to an adjacent empty space, which becomes tissue. As the initial condition we consider only region R1 occupied by tissue, while the remaining regions are empty. Processes beginning with *Empty* represent empty regions, with $Empty0$ denoting that region R0 is empty. Processes beginning with *Tissue* represent tissue regions, with $Tissue0$ denoting that region R0 contains tissue. Actions beginning with *apo* can be performed by tissue processes to change into empty space, representing cell death. Actions beginning with *mito* can be performed by a tissue process in synchronisation with an empty space process. As a result the empty space process involved is converted into a tissue process. Actions beginning with *mito* have a direction, from tissue to empty space. For example, $mito12$ can only be performed in synchronisation by $Tissue1$ and $Empty2$. Finally, the definition of the processes is:

$$Empty0 \triangleq mito10.Tissue0$$
$$Empty1 \triangleq mito01.Tissue1 + mito21.Tissue1$$
$$Empty2 \triangleq mito12.Tissue2$$
$$Tissue0 \triangleq apo0.Empty0 + mito01.Tissue0$$
$$Tissue1 \triangleq apo1.Empty1 + mito10.Tissue1 + mito12.Tissue1$$
$$Tissue2 \triangleq apo2.Empty2 + mito21.Tissue2$$

Process $Empty1$ is illustrated in Figure 3.4. The initial state of the model is defined by the following process:

$$(Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1) \underset{\mathcal{K}}{\bowtie} Empty2$$

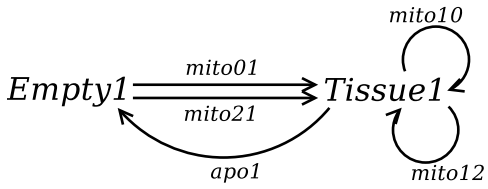


Figure 3.4: Graphical representation of process $Empty1$.

where $\mathcal{L} = \{mito10, mito01\}$ and $\mathcal{K} = \{mito12, mito21\}$. Examples of valid derivations are:

$$(Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1) \underset{\mathcal{K}}{\bowtie} Empty2 \xrightarrow{apo1} (Empty0 \underset{\mathcal{L}}{\bowtie} Empty1) \underset{\mathcal{K}}{\bowtie} Empty2$$

$$(Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1) \underset{\mathcal{K}}{\bowtie} Empty2 \xrightarrow{mito12} (Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1) \underset{\mathcal{K}}{\bowtie} Tissue2$$

It should also be noted that the following transition is prevented:

$$(Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1) \underset{\mathcal{K}}{\bowtie} Tissue2 \xrightarrow{mito12}$$

This is because, although $Tissue1$ could perform $mito12$, $mito12 \in \mathcal{K}$, $Tissue2$ cannot provide $mito12$ for synchronisation. In other words, neither derivation rule **Coop Left** nor **Synchronisation** are applicable.

We will return to this model when we will discuss a multi-scale model of tissue growth (Section 4.2.3).

Implicit Modelling of Empty Space. An alternative is the definition of a tissue process that can self replicate, without the need to synchronise with empty space processes. An example of this approach is given by the following process definition:

$$Tissue \triangleq apo.nil + mito.(Tissue \underset{\emptyset}{\bowtie} Tissue)$$

The initial state of the model could be defined simply as $Tissue$. The number of regions that are turned into tissue is given by the number of Tissue processes in the model. However, there are two issues concerning this approach:

1. following action apo, representing tissue death, a nil process is introduced which is cumbersome to remove and which may block actions of other processes;

2. if relative positions of regions have to be considered, then growth in an infinitely large space cannot be modelled and confined growth produces a combinatorial explosion of the definition of the model.

Consider the two points in turn:

1. for example, consider the above definition of the *Tissue* process and the following transition:

$$Tissue \underset{\emptyset}{\bowtie} Tissue \xrightarrow{apo} Tissue \underset{\emptyset}{\bowtie} nil$$

It is clear that in the long run *nil* processes will accumulate. Thus, it is desirable to replace $Tissue \underset{\emptyset}{\bowtie} nil$ with *Tissue*. In this example, this is appropriate, because $P \underset{\mathcal{L}}{\bowtie} nil$ can be substituted by P if P and all processes P' that can be reached via valid transitions from P cannot perform actions in $\mathcal{L}$. However, in general we may want new regions of tissue to communicate with existing ones, implying that $\mathcal{L}$ should contain actions used for such communication. As a consequence, it would be difficult to remove $\underset{\mathcal{L}}{\bowtie} nil$, because P and $P \underset{\mathcal{L}}{\bowtie} nil$ are in general not equivalent. For example consider the following definition of process *Tissue*:

$$Tissue \triangleq a.Tissue + apo.nil + mito.(Tissue \underset{\{a\}}{\bowtie} Tissue)$$

This represents tissue where action a is executed by all tissue regions together. Now consider the following transition:

$$Tissue \underset{a}{\bowtie} Tissue \xrightarrow{apo} Tissue \underset{a}{\bowtie} nil$$

This transition produces a state where $\underset{a}{\bowtie} nil$ is blocking action a, but we cannot remove it because $Tissue \underset{a}{\bowtie} nil$ and *Tissue* are not equivalent.

2. in order to distinguish which regions of tissue are adjacent, we need to extend this model, using different names for processes representing tissue in different regions. On the one hand, if we want tissue to be able to grow indefinitely, we have to specify an infinite number of processes. On the other hand, if a confined space composed of a finite number of regions is considered, a model can be defined, but its definition presents a combinatorial problem. Without loss of generality, we can define a self replicating tissue on an horizontal line of regions as follows, where $Tissue1$ represents tissue in region R1 (Figure 3.5):

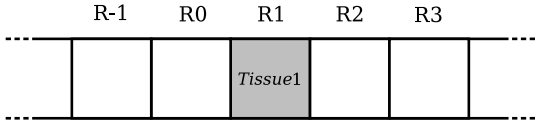


Figure 3.5: Implicit modelling of empty space on a line. Each region has a position identified by a natural number.

$$Tissue1 \triangleq apo1.nil + mito12.(Tissue1 \underset{\emptyset}{\bowtie} Tissue2) \\ + mito10.(Tissue1 \underset{\emptyset}{\bowtie} Tissue0))$$

The initial state of the model is given by the process $Tissue1$ alone. It is clear that this definition is not appropriate, because it may lead to multiple copies of process $Tissue2$. For example:

$$Tissue1 \xrightarrow{mito12} Tissue1 \underset{\emptyset}{\bowtie} Tissue2 \xrightarrow{mito12} \\ (Tissue1 \underset{\emptyset}{\bowtie} Tissue2) \underset{\emptyset}{\bowtie} Tissue2$$

In order to overcome this problem, the definition of $Tissue1$ needs to encode whether adjacent tissue is present or not. For example, if $Tissue1$ replicates adding process $Tissue2$, $Tissue1$ has to be updated to avoid the addition of another $Tissue2$ process. But this is not enough. If $Tissue3$ is present in the model, it should be updated as well because it is adjacent to $Tissue2$. Moreover, a similar neighbour update has to be performed when action $apo1$ is performed.

Unfortunately, there is no trivial solution to this problem. We illustrate why with an example. Assume we can define a $Tissue1$ (and analogous definitions for $Tissue2$, $Tissue3$...) process as follows:

$$Tissue1 \triangleq apo1.nil + mito12.(Tissue1' \underset{\mathcal{L}}{\bowtie} Tissue2)$$
$$+mito10.(Tissue1'' \underset{\mathcal{K}}{\bowtie} Tissue0 + mito\text{-}10.Tissue1''$$
$$+mito32.Tissue1')$$

$$Tissue1' \triangleq apo1.nil + apo2.Tissue1$$
$$+mito10.(Tissue1''' \underset{\mathcal{K}}{\bowtie} Tissue0) + mito\text{-}10.Tissue1'''$$

$$Tissue1'' \triangleq apo1.nil + apo0.Tissue1$$
$$+mito12.(Tissue1''' \underset{\mathcal{L}}{\bowtie} Tissue2) + mito32.Tissue1'''$$

$$Tissue1''' \triangleq apo1.nil + apo2.Tissue1'' + apo0.Tissue1'$$

The four $Tissue1$ processes represent different action capabilities of a region depending on the surrounding regions. $Tissue1$ expresses the actions available when regions are empty on the left and on the right, $Tissue1'$ when tissue is present only on the right, $Tissue1''$ when tissue is present only on the left and $Tissue1'''$ when tissue is surrounding the current region. However, this solution is not correct. The reason is that since, for example, $Tissue1$ is responsible for the addition of $Tissue2$, it should also be able to determine which version of $Tissue2$ has to be introduced, choosing between the four stated above and depending on whether $Tissue3$ is present or not. In fact, because of this mechanism, if a model were composed of $Tissue1$ and $Tissue100$, the behaviour of $Tissue1$ would depend on the fact that $Tissue100$ is present in the model. This implies a combinatorial explosion of process definitions.

Another problem is the choice of cooperation sets $\mathcal{L}$ and $\mathcal{K}$. For example, $apo(i)$ should be placed in both $\mathcal{L}$ and $\mathcal{K}$, in order to communicate to adjacent tissue that region i is empty again. However, once a $Tissue$ process becomes nil, there is no trivial way to remove it, blocking all the other parametric actions in the two cooperation sets.

We can conclude that, because of the problems discussed in this section, in the context of modelling tissue growth with multi-way synchronisation, the explicit modelling approach of empty space is preferable to the implicit modelling approach of empty space.

3.2 Stochastic Semantics for Simple Process Algebra

In this section we define a stochastic semantics based on functional rates for SPA. The motivation for functional rates comes from the fact that rates of biological events often depend on the current state of the system itself. More precisely, only a part of a biological system contributes to the determination of the rate of an event. In terms of process algebra, this means that an action is associated with a set of processes and these processes are associated with variables and values that are used to evaluate functional rates. An example is the evaluation of a rate for a biochemical event. Processes represent the concentration of species (the variables) and the current concentration level (the values). Functional rates based on kinetic laws are associated with actions. When processes synchronise via a specific action, the corresponding functional rate is evaluated according to the information associated to the processes.

In order to obtain a stochastic semantics for SPA, with the characteristics outlined above, we define:

- a new syntax for SPA, which should guarantee that at any time variables and values can be associated with processes (Section 3.2.1);
- a new semantics that determines which actions are valid along with collecting variables and values from the processes that participate in the actions. In particular we use functions $Var : \mathbb{P}_m \to Names$ and $Val : \mathbb{P}_m \to \mathbb{R}$, with $Names$ the set of parameter names. Variables and values are stored into an environment Γ. We define Γ as a partial function, with $\Gamma : Names \to \mathbb{R}$. We assume that Γ can be represented as a set of pairs of the form $(n, \Gamma(n)) \in Names \times \mathbb{R}$. Moreover, we assume that Γ can be extended using set union. The union $\Gamma_1 \cup \Gamma_2$ is well defined only if for all $a, b \in Names$, if $(a, x) \in \Gamma_1$ and $(b, y) \in \Gamma_2$ then $a \neq b$. The semantics yields labelled transitions with pairs (a, Γ) as labels, where a is an action and Γ an environment (Section 3.2.1);
- a syntax for functional rates (Section 3.2.2);

- a semantics for functional rates, that, given an environment determines valid evaluations of a functional rate to an actual rate, i.e. a real number (Section 3.2.2);

- a mechanism to decide when a functional rate can or cannot be evaluated for a specified pair (a, Γ). This evaluation is performed on a derivation graph generated by the semantics of the algebra and produces a rated derivation graph, where rated labels have the form (a, r), with $r \in \mathbb{R}^{>0}$, while labels that cannot be rated are unchanged (Section 3.2.4).

We illustrate some of the concepts just discussed with an example. Consider the following biochemical reaction R_a between biochemical species **A**, **B** and **C** and corresponding velocity v_a:

$$R_a : \mathbf{A} + \mathbf{B} \rightarrow_{v_a} \mathbf{C}, \qquad v_a = k_a[\mathbf{A}][\mathbf{B}]$$

The velocity v_a expresses the amount of concentration of **A** and **B** that is consumed per time unit. The symbol $[\cdot]$ means concentration, e.g. $[\mathbf{A}]$ is the concentration of species **A**, while k_a is a constant. To represent the interactions of R_a, we use the following processes:

$$\begin{array}{lll} A_0 \triangleq nil & B_0 \triangleq nil & C_0 \triangleq a.C_1 \\ A_1 \triangleq a.A_0 & B_1 \triangleq a.B_0 & C_1 \triangleq a.C_2 \\ A_2 \triangleq a.A_1 & B_2 \triangleq a.B_1 & C_2 \triangleq nil \end{array}$$

We use the *processes as levels of concentration* abstraction, in analogy with Section 3.2. The three processes for each species represent a different concentration level, from 0 to 2. The maximum concentration is fixed to M, while N is the maximum number of levels, here 2. The concentration represented by one level is given by h, with $h = M/N$.

As explained above, we associate a variable name and a value to each of the processes. To do so, we use functions $Var(\cdot)$ and $Val(\cdot)$. In particular we need to provide the following information:

$$
\begin{array}{lll}
Var(A_0) = levelA & Var(B_0) = levelB & Var(C_0) = levelC \\
Val(A_0) = 0 & Val(B_0) = 0 & Val(C_0) = 0 \\
Var(A_1) = levelA & Var(B_1) = levelB & Var(C_1) = levelC \\
Val(A_1) = 1 & Val(B_1) = 1 & Val(C_1) = 1 \\
Var(A_2) = levelA & Var(B_2) = levelB & Var(C_2) = levelC \\
Val(A_2) = 2 & Val(B_2) = 2 & Val(C_2) = 2
\end{array}
$$

The velocity v_a is used to produce the functional rate associated with action a in analogy with that for CTMC with levels (Section 2.2.8):

$$f_a = (k_a \cdot levelA \cdot h \cdot levelB \cdot h)/h$$

This means that in order to evaluate f_a we need to provide an additional environment containing values for constants h and k_a. This could be $\Gamma = \{(h, 1), (k_a, 1)\}$. Now assume that the initial state of the model is given by:

$$A_2 \underset{a}{\bowtie} (B_2 \underset{a}{\bowtie} C_0)$$

Intuitively, an appropriate stochastic semantics should permit the following transition:

$$A_2 \underset{a}{\bowtie} (B_2 \underset{a}{\bowtie} C_0) \xrightarrow{(a,r_a)} A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)$$

with $r_a = 4$, resulting from the evaluation of f_a.

We propose that the aim of a stochastic semantics should be to determine if an action a is possible and, at the same time, collect information about the variables and values associated with the processes that synchronise on a. For example, a valid transition is:

$$A_2 \underset{a}{\bowtie} (B_2 \underset{a}{\bowtie} C_0) \xrightarrow{(a,\Gamma')} A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)$$

with $\Gamma' = \{(levelA, 2), (levelB, 2), (levelC, 0)\}$. We call the pair (a, Γ') an *activity*. We can *rate* activity (a, Γ') using Γ', f_a and the additional constant environment Γ containing values of variables h and k_a. The result of the rating of (a, Γ') is the *rated activity* (a, r_a).

3.2.1 Stochastic Simple Process Algebra

In order to define a suitable stochastic semantics for SPA, we first modify the syntax. In particular, we want to make sure that variables and values can always be associated with processes in a model. In order to do so, we associate variables and values only with agents and restrict the model definition to cooperations of agents. This means that agents now play a central role in the algebra. Thus, we divide the syntax into *definition* processes (D) and *model* processes (M) as follows:

$$D ::= nil \mid a.A \mid D + D$$
$$M ::= A \mid M \underset{\mathcal{L}}{\bowtie} M$$

where:

- D is a *definition* process, $D \in \mathbb{P}_d$, while M is a *model* process, $M \in \mathbb{P}_m$. Definition and model processes are disjoint and are both processes, i.e. $\mathbb{P}_d \cup \mathbb{P}_m = \mathbb{P}$ and $\mathbb{P}_d \cap \mathbb{P}_m = \emptyset$;
- agent A is defined as $A \triangleq D$, that is we use definition processes to define the behaviour of agents, restricting agent definitions to choices of actions;
- a model is defined by a model process M, which in turn is either an agent A or a cooperation between model processes $M \underset{\mathcal{L}}{\bowtie} M$;
- action execution $a.A$ is always followed by an agent A. This ensures that at any time the state of a model will consist of a cooperation of agents;
- each agent A is associated with a variable and a value, that is functions $Var(A) \in Names$, with $Names$ the set of parameter names, and $Val(A) \in \mathbb{R}$ must be defined for all agents A. This, along with the above definitions, ensures that variables and values can always be associated with processes in a model.

The stochastic semantics for this new syntax is shown in Figure 3.6. With this semantics, information of variables and values is collected in the set Γ. In the derivation rule **Synchronisation**, union of environments Γ_1 and Γ_2 is valid only if for all $a, b \in Names$, if $(a, x) \in \Gamma_1$ and $(b, y) \in \Gamma_2$ then $a \neq b$. The

Prefix

$$\frac{}{a.A \xrightarrow{a} A}$$

Choice Left

$$\frac{D_1 \xrightarrow{a} A}{D_1 + D_2 \xrightarrow{a} A}$$

Choice Right

$$\frac{D_2 \xrightarrow{a} A}{D_1 + D_2 \xrightarrow{a} A}$$

Coop Left

$$\frac{M_1 \xrightarrow{(a,\Gamma)} M_1'}{M_1 \underset{\mathcal{L}}{\bowtie} M_2 \xrightarrow{(a,\Gamma)} M_1' \underset{\mathcal{L}}{\bowtie} M_2} \text{ if } a \notin \mathcal{L}$$

Coop Right

$$\frac{M_2 \xrightarrow{(a,\Gamma)} M_2'}{M_1 \underset{\mathcal{L}}{\bowtie} M_2 \xrightarrow{(a,\Gamma)} M_1 \underset{\mathcal{L}}{\bowtie} M_2'} \text{ if } a \notin \mathcal{L}$$

Synchronisation

$$\frac{M_1 \xrightarrow{(a,\Gamma_1)} M_1' \quad M_2 \xrightarrow{(a,\Gamma_2)} M_2'}{M_1 \underset{\mathcal{L}}{\bowtie} M_2 \xrightarrow{(a,\Gamma_1 \cup \Gamma_2)} M_1' \underset{\mathcal{L}}{\bowtie} M_2'} \text{ if } a \in \mathcal{L}$$

Agent

$$\frac{D \xrightarrow{a} A'}{A \xrightarrow{(a,\Gamma)} A'} \text{ if } A \triangleq D \wedge \Gamma = \{(Var(A), Val(A))\}$$

Figure 3.6: Stochastic semantics of a simple process algebra.

collected environments will be used in a second moment to computed the rates of the transitions, producing a *rated* derivation graph.

We refer to this new process algebra as *stochastic simple process algebra* (sSPA). A valid derivation for the running example in Section 3.2 is the following transition:

$$A_2 \underset{a}{\bowtie} (B_2 \underset{a}{\bowtie} C_0) \xrightarrow{(a,\Gamma)} A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)$$

where $\Gamma = \{(levelA, 2), (levelB, 2), (levelC, 0)\}$. In general, the stochastic semantics produces a derivation graph consisting of an initial state, the reachable states and valid transitions. In order to define this formally we introduce the following definitions.

Definition 3.1 *Activity.* The pair (a, Γ) such that $a \in Actions$ and $\Gamma \subseteq Names \times \mathbb{R}$ is called an activity.

Definition 3.2 *One step derivative.* If $M \xrightarrow{(a,\Gamma)} M'$ then M is a one step derivative of M.

Definition 3.3 *Derivative.* If $M_i \xrightarrow{(a,\Gamma)} \ldots \xrightarrow{(a',\Gamma')} M_j$ then M_j is a derivative of M_i.

Definition 3.4 *Derivative Set.* The derivative set of a model process $M \in \mathbb{P}_m$ is denoted by $ds(M)$ and is defined as the smallest set of model processes such that:

- $M \in ds(M)$;
- if $M_i \in ds(M)$ and $M_i \xrightarrow{(a,\Gamma)} M_j$ then $M_j \in ds(M)$.

Definition 3.5 *Current actions for definition processes.* The set of actions that $D \in \mathbb{P}_d$ can perform is denoted by $Actions(D)$ and is defined as:

- $Actions(nil) = \{\!|\}$;
- $Actions(a.A) = \{\!|a|\!\}$;
- $Actions(D_1 + D_2) = Actions(D_1) \uplus Actions(D_2)$.

with $\{\!|\}$ delimiting a multi-set and $\uplus$ the sum of multi-sets.

Definition 3.6 *Current activities for model Processes.* The set of activities that $M \in \mathbb{P}_m$ can perform is denoted by $Activities(M)$ and is defined as:

- $Activities(A) = \{\!|(a,\Gamma) \mid a \in Actions(D) \wedge A \triangleq D \wedge \Gamma = \{(\,Var(A),\, Val(A))\}|\!\}$;
- $Activities(M_1 \underset{\mathcal{L}}{\bowtie} M_2) =$
  $\{\!|(a,\Gamma) \mid (a,\Gamma) \in Activities(M_1) \wedge a \notin \mathcal{L}|\!\}$
  $\uplus\, \{\!|(a,\Gamma) \mid (a,\Gamma) \in Activities(M_2) \wedge a \notin \mathcal{L}|\!\}$
  $\uplus \{\!|(a,\Gamma_1 \cup \Gamma_2) \mid (a,\Gamma_1) \in Activities(M_1) \wedge (a,\Gamma_2) \in Activities(M_2) \wedge a \in \mathcal{L}|\!\}$.

Definition 3.7 *Activity set.* The set of all activities that a model process $M \in \mathbb{P}_m$ or one of its derivatives can perform is given by:

$$\overrightarrow{Activities}(M) = \biguplus_{M_i \in ds(M)} Activities(M_i)$$

Definition 3.8 *Derivation graph.* Given a model component $M \in \mathbb{P}_m$, the derivation graph $\mathcal{D}(M)$ is the labelled directed graph with:

- set of nodes $ds(M)$;
- multi-set of transition labels $\overrightarrow{Activities}(M)$;
- multi-set of labelled transitions $\rightarrow \subseteq ds(M) \times \overrightarrow{Activities}(M) \times ds(M)$. Given $M' \in ds(M)$, $(M', a, \Gamma, M'') \in \rightarrow$ iff $M' \xrightarrow{(a,\Gamma)} M''$.

The derivation graph of the running example of Section 3.2, $\mathcal{D}(A_2 \underset{a}{\bowtie} (B_2 \underset{a}{\bowtie} C_0))$ is:

$$A_2 \underset{a}{\bowtie} (B_2 \underset{a}{\bowtie} C_0) \xrightarrow{(a,\Gamma)} A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1) \xrightarrow{(a,\Gamma')} A_0 \underset{a}{\bowtie} (B_0 \underset{a}{\bowtie} C_2)$$

with $\Gamma = \{(levelA, 2), (levelB, 2), (levelC, 0)\}$ and $\Gamma' = \{(levelA, 1), (levelB, 1), (levelC, 1)\}$.

Before we can introduce the formal definition of a *rated* derivation graph, we introduce the syntax of functional rates and semantics for valid evaluations.

3.2.2 Formalisation of Functional Rates

We introduce now a formal definition of functional rates and their evaluation. The syntax of functional rates is given by:

$$f ::= k \mid i \mid f\ op_1\ f \mid op_2(f) \mid f^f$$

$$op_1 ::= + \mid - \mid * \mid / \qquad op_2 ::= \exp \mid \log \mid \sin \mid \cos$$

where:

- $k \in \mathbb{R}$;
- $i \in Names$, i.e. i is a parameter name;
- f is a functional rate, $f \in \mathbb{F}$, which essentially is an arithmetical expression, with operators op_1 and op_2;
- op_1 are unary operators, while op_2 are binary operators.

Constant

$$\frac{}{\Gamma \vdash k \to k} \; k \in \mathbb{R}$$

Variable

$$\frac{}{\Gamma \vdash i \to k} \; \Gamma(i) = k$$

Unary Operator

$$\frac{\Gamma \vdash exp \to k_1}{\Gamma \vdash op_2(exp) \to k_2} \; k_2 = op_2(k_1)$$

Binary Operator

$$\frac{\Gamma \vdash exp_1 \to k_1 \qquad \Gamma \vdash exp_2 \to k_2}{\Gamma \vdash exp_1 \; op_1 \; exp_2 \to k_3} \; k_3 = k_1 \; op_1 \; k_2$$

Exponential Operator

$$\frac{\Gamma \vdash exp_1 \to n_1 \qquad \Gamma \vdash exp_2 \to n_2}{\Gamma \vdash exp_1^{exp_2} \to n_3} \; n_3 = n_1^{n_2}$$

Figure 3.7: Semantics for the evaluation of functional rates.

In order to evaluate functional rates expressed with this syntax, we use the semantics in Figure 3.7. This is in fact a standard operational semantics for arithmetical expressions. Given an environment $\Gamma \subseteq Names \times \mathbb{R}$ and a functional rate f, f evaluates to k if $\Gamma \vdash f \to k$ is a valid derivation. It is of course possible to add other constant values such as the step size h or kinetic constants to the environment Γ before the evaluation of the functional rate. We illustrate this with an example. Recall the functional rate we defined earlier, $f_a = (k_a \cdot levelA \cdot h \cdot levelB \cdot h)/h$. The environment Γ we derived for a possible transition is $\Gamma = \{(levelA, 2), (levelB, 2), (levelC, 0)\}$. With the addition of environment $\Gamma' = \{(k_a, 1), (h, 1)\}$, it follows that f evaluates to 4, because $\Gamma'' \vdash f \to 4$, $\Gamma'' = \Gamma \cup \Gamma'$, with the derivation in Figure 3.8.

3.2.3 Normalisation

In this section we derive a well formed definition of sSPA processes that guarantees that the rates are computed correctly. In addition, we introduce the concept of *set of participants*. An action a is associated with a functional rate f_a if and only if it is associated with a set of participants p_a. This latter indicates which variables, and so which entities participate to the biological event represented by

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma'' \vdash k_a \to 1} \quad \dfrac{}{\Gamma'' \vdash levelA \to 2}}{\Gamma'' \vdash k_a * levelA \to 2} \quad \dfrac{}{\Gamma'' \vdash h \to 1}}{\Gamma'' \vdash k_a * levelA * h \to 2} \quad \dfrac{}{\Gamma'' \vdash levelB \to 2}}{\Gamma'' \vdash k_a * levelA * h * levelB \to 4} \quad \dfrac{}{\Gamma'' \vdash h \to 1}}{\Gamma'' \vdash k_a * levelA * h * levelB * h \to 4} \quad \dfrac{}{\Gamma'' \vdash h \to 1}}{\Gamma'' \vdash (k_a * levelA * h * levelB * h)/h \to 4}$$

Figure 3.8: Example of a derivation for a valid evaluation of a functional rate.

action a. These variables are also sufficient to evaluate f_a, though not all of them may be necessary. In fact, p_a is not just the set of variables used to evaluate f_a, it is used to represent the scope of action a, and so to identify if all processes associated with variables in p_a that are assumed to synchronise on a have done so. This mechanism is the key to the proof of congruence of the equivalence relations we introduce in Chapter 6.

In some cases the rate r_a computed from a functional rate f_a associated with action a is not correct and requires to be scaled. Namely, the rate computed is divided by the number of actions a a process can perform. This procedure is referred to as *normalisation* in (Bernardo, 1996), and usually needs to be applied when a single rate is associated with multiple actions, a situation often caused by non deterministic choices. The well formed definition ensures normalisation is performed correctly and is sound with respect to biological assumptions.

In sSPA as defined so far, we identify two sources of non deterministic choice that require particular attention, one generated by the + operator and the other by the $\underset{\mathcal{L}}{\bowtie}$ operator. The first type of non deterministic choice always requires normalisation, while the second type requires normalisation only in some cases and might interfere with the normalisation of the first. To illustrate this problem and motivate the solution, consider the following four processes, recalling the agent definitions of our running example for this section:

1. $A_1' \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)$
2. $A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{\emptyset}{\bowtie} C_1))$
3. $A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{\emptyset}{\bowtie} D_1))$
4. $(A_1 \underset{\emptyset}{\bowtie} A_1) \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)$

Additional agent definitions and associated variables and values are:

$$A_1' \triangleq a.A_0 + a.A_2 \quad Var(A_1') = levelA \quad Val(A_1') = 1$$
$$D_1 \triangleq a.D_2 \quad Var(D_1) = levelD \quad Val(D_1) = 1$$

The current activities of the four processes are:

1. $Activities(A_1' \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)) = \{\!|(a,\Gamma),(a,\Gamma)|\!\}$
2. $Activities(A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{\emptyset}{\bowtie} C_1))) = \{\!|(a,\Gamma),(a,\Gamma)|\!\}$

3. $Activities(A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{\emptyset}{\bowtie} D_1))) = \{\!|(a,\Gamma),(a,\Gamma')|\!\}$

4. $Activities((A_1 \underset{\emptyset}{\bowtie} A_1) \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)) = \{\!|(a,\Gamma),(a,\Gamma)|\!\}$

where $\Gamma = \{(levelA,1),(levelB,1),(levelC,1)\}$ and $\Gamma' = \{(levelA,1),(levelB,1),(levelD,1)\}$. Recall that the functional rate associated with a is:

$$f_a = (k_a \cdot levelA \cdot h \cdot levelB \cdot h)/h$$

The activities can then be rated using f_a and the additional set $\Gamma'' = \{(h,1),(k_a,1)\}$. We can assume (a,Γ) is rated to $(a,1)$, because $\Gamma \cup \Gamma'' \vdash f_a \rightarrow 1$ and (a,Γ') is also rated to $(a,1)$, because $\Gamma' \cup \Gamma'' \vdash f_a \rightarrow 1$. Thus, assuming $RatedAct(M)$ is the multi-set of rated activities of process M:

1. $RatedAct(A_1' \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)) = \{\!|(a,1),(a,1)|\!\}$

2. $RatedAct(A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{\emptyset}{\bowtie} C_1))) = \{\!|(a,1),(a,1)|\!\}$

3. $RatedAct(A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{\emptyset}{\bowtie} D_1))) = \{\!|(a,1),(a,1)|\!\}$

4. $RatedAct((A_1 \underset{\emptyset}{\bowtie} A_1) \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)) = \{\!|(a,1),(a,1)|\!\}$

In the case of process 1, the two rated activities $(a,1)$ and $(a,1)$ should be normalised, that is the actual rated activities should be divided by the multiplicity of (a,Γ), in this case 2. This is because, under the non deterministic choice caused by the + operator in A_1', we assume that the rate r_a evaluated from the functional rate f_a, represents the *total exit rate* for action a. This assumption is based on the fact that the two a actions represent two different outcomes of the same biological event involving the same biological species. Thus, the actual multi-set of rated activities for process 1 should be:

$$RatedAct(A_1' \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1)) = \{\!|(a,0.5),(a,0.5)|\!\}$$

If the deterministic choice is generated by the $\underset{\mathcal{L}}{\bowtie}$ operator, as in processes 2, 3 and 4, it becomes more complicated to determine whether normalisation should be applied. It usually depends on what processes and actions represent, and what functional rate is associated with the actions.

In the case of process 2, the two identical rated activities $(a,1)$ and $(a,1)$ should also be normalised. This is because the non deterministic choice is between two agents C_1 which represent the concentration of biochemical species **C**, which

in turn does not contribute to the determination of the rate. The concentrations of **A** and **B** determine the exit rate for action a, while the produced concentration of **C** can be added non deterministically to one of the pools of concentration represented by the two agents C_1. Thus, the actual multi-set of rated activities for process 2 should be:

$$RatedAct(A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{\emptyset}{\bowtie} C_1))) = \{\!|(a, 0.5), (a, 0.5)|\!\}$$

In the case of process 3, the two rated activities $(a, 1)$ and $(a, 1)$ should also be normalised. This should be done even if the environment of those activities differ, as in this case, where one activity has environment Γ, while the other Γ'. The reason is analogous to that for process 2. The total exit rate is determined by the concentration of **A** and **B**, while the produced concentration can be the concentration of **C** *or* of **D**. The actual multi-set of rated activities for process 3 should be:

$$RatedAct(A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{\emptyset}{\bowtie} D_1))) = \{\!|(a, 0.5), (a, 0.5)|\!\}$$

In the case of process 4, the two rated activities $(a, 1)$ and $(a, 1)$ should *not* be normalised. This is because the non deterministic choice is between two agents A_1 that represent the concentration of biochemical species **A**, which in turn contributes to the determination of the rate. More precisely, we can observe that the rate of action a is proportional to the concentration level represented by A_1. Each agent A_1 represents a certain "pool" of concentration of species **A**, which can interact independently with species **B** and **C**. The total concentration of **A** present in the system is the sum of the concentration represented by the two agents A_1. Since the rate is proportional to the concentration of **A**, then the total exit rate should be given by the sum of the two individual rates.

We have seen that for processes 1, 2, 3 and 4 the multi-set of rated activities is $\{\!|(a, 1), (a, 1)|\!\}$ if normalisation is not applied. We have also seen that normalisation is applied only in some cases, which might be difficult to identify. In particular, without the knowledge of what process performed the actions and what type of non determinism is involved, we are not able to determine whether normalisation is necessary or not. In order to have an automatic procedure for normalisation, we propose that the non-determinism caused by $\underset{\mathcal{L}}{\bowtie}$ operator is not valid. This can be enforced easily if:

- in every state of the system there is only one process associated with a certain variable. This can be ensured by insisting that the initial state has this property and that for all agents A, whenever $A \xrightarrow{(a,\Gamma)} A'$ then $Var(A) = Var(A')$, conditions that can be tested on the definition of the model;

- an action a is associated with a set of participants $p_a \subseteq Names$, in this case $p_a = \{levelA, levelB, levelC\}$. An activity (a, Γ) can be rated only if the variables contained in Γ are exactly the variables contained in p_a. This ensures that if there is a non deterministic choice between agents with different associated variables, only one will be considered a valid choice for rate evaluation.

With these constraints, it is easy to see that processes 2 and 4 are *not well formed*, because they present two processes with the same associated variable, that is $levelC$ in the first case and $levelA$ in the second. Moreover, in the case of process 3, activity (a, Γ') cannot be evaluated to $(a, 1)$ because Γ' does not contain exactly the variables contained in p_a.

Finally, the use of a set of participants p_a raises the question of what happens if processes associated with variables not in p_a synchronise on a. This could have the effect of turning an activity that can be rated (variables in Γ equal to p_a) into one that cannot be rated. For example, consider the following model process:

5. $A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{a}{\bowtie} D_1))$

The corresponding set of current activities is:

5. $Activities(A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} (C_1 \underset{a}{\bowtie} D_1))) = \{\!|(a, \Gamma'')|\!\}$

where $\Gamma'' = \{(levelA, 1), (levelB, 1), (levelC, 1), (levelD, 1)\}$. Activity (a, Γ'') cannot be rated, because the variables in Γ'' are not exactly the variables in p_a. To prevent this situation we add the following constraint:

- if f_a is a functional rate in $\mathbb{F}$ then a derivative of process P can perform action a if and only if $Var(P)$ is in p_a.

We can now define an sSPA model and a well formed sSPA model:

Definition 3.9 sSPA *model.* An sSPA model is a tuple:

$$(AgentDef, M, Actions, Names, \mathbb{F}, \Gamma, Participants, Var, Val)$$

where:

- *AgentDef* is the finite set of agent definitions $\{A_1 \triangleq D_1,\ A_2 \triangleq D_2, \dots\}$;
- M is the initial state of the model, with $M \in \mathbb{P}_m$;
- *Actions* is the finite set of actions;
- $Names$ is the finite set of parameter names;
- $\mathbb{F}$ is the finite set of functional rates;
- Γ is the finite set of constant model parameters, with $\Gamma \subseteq Names \times \mathbb{R}$;
- $Participants$ is the finite set of sets of participants;
- *Var* and *Val* are the functions associating agents with variables (i.e. parameter names) and values, with $Var : \mathbb{P}_m \to Names$ and $Val : \mathbb{P}_m \to \mathbb{R}$.

Definition 3.10 *Well formed* sSPA *model.* An sSPA model is well formed if and only if:

1. Given a model process as a cooperation of agents, of the form

$$A_1 \underset{\mathcal{L}_1}{\bowtie} A_2 \underset{\mathcal{L}_2}{\bowtie} \dots \underset{\mathcal{L}_{n-1}}{\bowtie} A_n$$

   then $\forall A_i, A_j$ if $i \neq j$ then $Var(A_j) \neq Var(A_j)$;

2. Given a definition process as a choice of sequential actions, of the form

$$A \triangleq \sum_i a_i.A_i$$

   then $\forall A_i\ Var(A) = Var(A_i)$;

3. $\forall a$ s.t. $f_a \in \mathbb{F}$

$$\exists (a, \Gamma) \in \overrightarrow{Activities}(P) \Leftrightarrow Var(P) \in p_a$$

3.2.4 The Rating Routines

In the previous sections we have defined *how* to evaluate a functional rate f using the environment Γ of an activity and an additional constant environment Γ'. In this section we define a mechanism to determine *when* it is appropriate to evaluate a functional rate and we determine what happens when an evaluation is not possible. The result is a rating procedure that converts a derivation graph (Definition 3.8) into a *rated* derivation graph.

Recall again our running example. If we consider a model composed only of process A_2, the following derivation is valid:

$$A_2 \xrightarrow{(a,\{(levelA,2)\})} A_1$$

Given the above transition and the functional rate f_a associated with action a, it is clear that an evaluation of f_a is not possible. In fact, environment $\Gamma = \{(levelA, 2)\}$ is missing variable $levelB$. In this case, we call the transition and the activity (a, Γ) *open*.

Now consider the following transition:

$$A_2 \underset{a}{\bowtie} B_2 \xrightarrow{(a,\Gamma')} A_1 \underset{a}{\bowtie} B_1$$

where $\Gamma' = \{(levelA, 2), (levelB, 2)\}$. In this case, f_a can be evaluated correctly, because all the necessary variables are included in Γ'. However, we consider this transition open as well and we do not allow the functional rate to be evaluated just yet. This is because not all the processes that will be affected by the action synchronise on a.

Formally, we consider the list of participants of action a, $p_a = \{levelA, levelB, levelC\}$. Participants are variables required for the evaluation of the rate to be successful. These may not only be variables whose values we need to know in order to evaluate the rate, but also variables associated with elements that will be affected by the reaction.

The concept of participants of an action has an important role in our approach to functional rates. Later on we will show how we exploit this concept in compositionality (Section 6). Intuitively, we use the list of participants to indicate exactly which processes will synchronise on a certain action. Once the environment Γ collected from the execution of action a contains all the variables in p_a, f_a can be evaluated and we assume no further processes will synchronise.

This is a quite strong assumption and it is based on the observation that, at least in biology, one should always be able to identify which elements in a system are affected by a certain event.

Finally, we know that some transitions can be turned into rated transitions, while others cannot. This implies that the derivation graph resulting after the *rating* operation will include two types of transitions, *open* transitions and *rated* transitions.

We proceed now to the formalisation of the procedure used to rate activities and the definition of a rated derivation graph.

Definition 3.11 *Function* envVar. The function *envVar* extracts the set of variables in an environment $\Gamma \subseteq Names \times \mathbb{R}$:

$$envVar(\Gamma) = (\{i \mid (i,k) \in \Gamma\})$$

Definition 3.12 *Open activity.* An open activity is an activity (a, Γ) where Γ does not contain the *exact* variables present in the participant set p_a, i.e. $p_a \neq envVar(\Gamma)$.

Definition 3.13 *Function* openActivities. The function *openActivities* selects open activities from a set of activities $A \subseteq Actions \times 2^{Names \times \mathbb{R}}$:

$$openActivities(A) = \left(\{\!|(a,\Gamma) \mid \ p_a \neq envVar(\Gamma) \wedge (a,\Gamma) \in A \ |\!\} \right)$$

Definition 3.14 *Current open activities.* Given a model process $M \in \mathbb{P}_m$, the set of open activities that P can perform is defined as:

$$OpenAct(M) = openActivities(Activities(M))$$

Definition 3.15 *Open activity set.* The set of all open activities that a model process $M \in \mathbb{P}_m$ can perform is given by:

$$\overrightarrow{OpenAct}(M) = openActivities(\overrightarrow{Activities}(M))$$

Definition 3.16 *Closed activity.* A closed activity is an activity (a, Γ) where Γ contains the exact variables present in the participant set p_a, i.e. $p_a = envVar(\Gamma)$.

Definition 3.17 *Function* closedActivities. The function *closedActivities* selects closed activities from a set of activities $A \subseteq Actions \times 2^{Names \times \mathbb{R}}$:

$$closedActivities(A) = (A \setminus openActivities(A))$$

Definition 3.18 *Current closed activities.* Given a model process $M \in \mathbb{P}_m$, the set of closed activities that M can perform is defined as:

$$ClosedAct(M) = closedActivities(Activities(M))$$

Definition 3.19 *Closed activity set.* The set of all open activities that a model process $M \in \mathbb{P}_m$ can perform is given by:

$$\overrightarrow{ClosedAct}(M) = closedActivities(\overrightarrow{Activities}(M))$$

Definition 3.20 *Rated activity.* The pair (a, r) such that $a \in Actions$ and $r \in \mathbb{R}_{>0}$ is called a rated activity.

Definition 3.21 *Function* rateActivities. Given an environment Γ, *rateActivities* converts a set of activities $A \subseteq Actions \times 2^{Names \times \mathbb{R}}$ into a set of rated activities $B \subseteq Actions \times \mathbb{R}$:

$$rateActivities(\Gamma)(A) =$$
$$\{\!|(a, r_a) \mid \Gamma \cup \Gamma' \vdash f_a \to k \wedge r_a = k/\pi(A, (a, \Gamma')) \wedge \ (a, \Gamma') \in A \ \wedge \ f_a \in \mathbb{F} \ |\!\}$$

where $\pi(A, (a, \Gamma'))$ returns the number of occurrences of (a, Γ') in the multi-set A.

Definition 3.22 *Current rated activities.* Given a model process $M \in \mathbb{P}_m$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the set of rated activities that M can perform is defined as:

$$RatedAct(M)_\Gamma = rateActivities(\Gamma)(ClosedAct(M))$$

$RatedAct(M)_\Gamma$ can be written $RatedAct(M)$ if Γ is clear from the context.

Definition 3.23 *Rated activity set.* Given an environment $\Gamma \subseteq Names \times \mathbb{R}$, the set of all rated activities that a model process $M \in \mathbb{P}_m$ can perform is given by:

$$\overrightarrow{RatedAct}(M)_\Gamma = rateActivity(\Gamma)(\overrightarrow{ClosedAct}(M))$$

$\overrightarrow{RatedAct}(M)_\Gamma$ can be written $\overrightarrow{RatedAct}(M)$ if Γ is clear from the context.

Definition 3.24 *Rated derivation graph.* Given a model process $M \in \mathbb{P}_m$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the rated derivation graph $\mathcal{D}_r(M)_\Gamma$ is the labelled directed graph with:

- set of nodes $ds(M)$;
- multi-set of transition labels $\overrightarrow{RatedAct}(M)_\Gamma$;
- multi-set of labelled transitions $\rightarrow_r \subseteq ds(M) \times \overrightarrow{RatedAct}(M)_\Gamma \times ds(M)$. Given $M' \in ds(M)$, $(M', a, r_a, M'') \in \rightarrow_r$ iff $M' \xrightarrow{(a,\Gamma')} M''$, $(a, \Gamma') \in ClosedAct(M)$ and $\{\!|(a,k)|\!\} = \text{rateActivities}(\Gamma)(\{\!|(a,\Gamma')|\!\})$ and $r_a = k/\pi(ClosedAct(M), (a, \Gamma'))$.
- multi-set of labelled transitions $\rightarrow_o \subseteq ds(M) \times \overrightarrow{OpenAct}(M) \times ds(M)$. Given $M' \in ds(M)$, $(M', a, \Gamma', M'') \in \rightarrow_o$ iff $M' \xrightarrow{(a,\Gamma')} M''$ and $(a, \Gamma') \in OpenAct(M)$.

$\mathcal{D}_r(M)_\Gamma$ can be written $\mathcal{D}_r(M)$ if Γ is clear from the context.

The rated derivation graph of the running example of Section 3.2, $\mathcal{D}_r(A_2 \underset{a}{\bowtie} (B_2 \underset{a}{\bowtie} C_0))_\Gamma$, with $\Gamma = \{(k_a, 1), (h, 1)\}$, is:

$$A_2 \underset{a}{\bowtie} (B_2 \underset{a}{\bowtie} C_0) \xrightarrow{(a,4)} A_1 \underset{a}{\bowtie} (B_1 \underset{a}{\bowtie} C_1) \xrightarrow{(a,1)} A_0 \underset{a}{\bowtie} (B_0 \underset{a}{\bowtie} C_2)$$

3.2.5 Stochastic Simple Process Algebra and Tissue Growth

We return now to the example we introduced in Section 3.1.2, with the addition of functional rates. In particular, we associate a constant rate k_{apo} to apoptosis, i.e. tissue death, and a rate k_{mito} to mitosis, i.e. tissue replication. In order to demonstrate that a rate can depend on the current state of the system, we assume that k_{mito} is the total rate of mitosis of a region containing tissue, that

is the sum of the rates of all mitosis actions from that region. This implies that functional rates of mitosis actions depend on whether adjacent regions contain tissue or not.

The definition of the processes is:

$$
\begin{aligned}
Empty0 &\triangleq mito10.Tissue0 + mito12.Empty0 \\
Empty1 &\triangleq mito01.Tissue1 + mito21.Tissue1 \\
Empty2 &\triangleq mito12.Tissue2 + mito10.Empty2 \\
Tissue0 &\triangleq apo0.Empty0 + mito01.Tissue0 + mito12.Tissue0 \\
Tissue1 &\triangleq apo1.Empty1 + mito10.Tissue1 + mito12.Tissue1 \\
Tissue2 &\triangleq apo2.Empty2 + mito21.Tissue2 + mito10.Tissue2
\end{aligned}
$$

The initial state of the model is defined by the following process:

$$Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1 \underset{\mathcal{K}}{\bowtie} Empty2$$

where $\mathcal{L} = \{mito10, mito01\}$ and $\mathcal{K} = \{mito12, mito21\}$. Variables and values associated to processes are:

$$
\begin{aligned}
Var(Empty0) = Var(Tissue0) &= region_0 \quad Val(Empty0) = 0 \quad Val(Tissue0) = 1 \\
Var(Empty1) = Var(Tissue1) &= region_1 \quad Val(Empty1) = 0 \quad Val(Tissue1) = 1 \\
Var(Empty2) = Var(Tissue2) &= region_2 \quad Val(Empty2) = 0 \quad Val(Tissue2) = 1
\end{aligned}
$$

Functional rates and sets of participants are defined as:

$$
\begin{aligned}
f_{mito01} &= k_{mito} & p_{mito01} &= \{region_0, region_1\} \\
f_{mito12} &= k_{mito}/(2 - region_0) & p_{mito12} &= \{region_0, region_1, region_2\} \\
f_{mito21} &= k_{mito} & p_{mito21} &= \{region_2, region_1\} \\
f_{mito10} &= k_{mito}/(2 - region_2) & p_{mito10} &= \{region_0, region_1, region_2\} \\
f_{apo0} &= k_{apo} & p_{apo0} &= \{region_0\} \\
f_{apo1} &= k_{apo} & p_{apo1} &= \{region_1\} \\
f_{apo2} &= k_{apo} & p_{apo2} &= \{region_2\}
\end{aligned}
$$

For example, f_{mito12} implies that if region R0 is empty then the rate for $mito12$ has to be divided by 2. The case is analogous for f_{mito10}. As a consequence $Tissue1$ performs mitosis always at a total rate of k_{mito}.

Examples of valid derivations are:

$$Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1 \underset{\mathcal{K}}{\bowtie} Empty2 \xrightarrow{(apo1,\Gamma)} Empty0 \underset{\mathcal{L}}{\bowtie} Empty1 \underset{\mathcal{K}}{\bowtie} Empty2$$

$$Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1 \underset{\mathcal{K}}{\bowtie} Empty2 \xrightarrow{(mito12,\Gamma')} Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1 \underset{\mathcal{K}}{\bowtie} Tissue2$$

where $\Gamma = \{(region_1, 1)\}$ and $\Gamma' = \{(region_0, 0), (region_1, 1), (region_2, 0)\}$. Using the additional environment $\Gamma'' = \{(k_{mito}, 1), (k_{apo}, 2)\}$, the transitions can be rated yielding:

$$Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1 \underset{\mathcal{K}}{\bowtie} Empty2 \xrightarrow{(apo1,2)} Empty0 \underset{\mathcal{L}}{\bowtie} Empty1 \underset{\mathcal{K}}{\bowtie} Empty2$$

$$Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1 \underset{\mathcal{K}}{\bowtie} Empty2 \xrightarrow{(mito12,0.5)} Empty0 \underset{\mathcal{L}}{\bowtie} Tissue1 \underset{\mathcal{K}}{\bowtie} Tissue2$$

3.3 Parametric Simple Process Algebra

We have seen in several examples so far that model definitions tend to be repetitive when modelling levels of concentration or entities located in space. In this section we show how the description of a model written in SPA can be reduced by the introduction of parameters and an "if then else" construct.

The syntax of the parametric simple process algebra with multi-way synchronisation (pSPA) is:

$$P ::= nil \mid a(exp, \ldots, exp).P \mid P + P \mid P \underset{\mathcal{L}}{\bowtie} P \mid [bexp]?P : P \mid A(exp, \ldots, exp)$$

$$exp ::= k \mid i \mid exp + exp \mid exp - exp \mid exp/k'$$

$$bexp ::= exp = exp \mid exp < exp \mid bexp \wedge bexp \mid bexp \vee bexp \mid \neg bexp \mid true \mid false$$

where:

- P is a process, $P \in \mathbb{P}$, with $\mathbb{P}$ the set of processes;
- nil is the deadlock process;
- a is an action, $a \in Actions$, with $Actions$ the set of actions;
- $exp_1, \ldots, exp_n$ is a non empty list of expressions;
- exp is an expression, which can be an real number k, $k \in \mathbb{R}$, a sum of two expressions, a difference between two expressions, a division between an expression and an real number $k' \in (\mathbb{R} \setminus \{0\})$, the parameter name $i \in Names$;

Prefix

$$\frac{}{a(exp_1,\ldots,exp_n).P \xrightarrow{a(exp_1,\ldots,exp_n),true} P}$$

Choice Left

$$\frac{P \xrightarrow{a(exp_1,\ldots,exp_n),b} P'}{P + Q \xrightarrow{a(exp_1,\ldots,exp_n),b} P'}$$

Coop Left

$$\frac{P \xrightarrow{a(exp_1,\ldots,exp_n),b} P'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{a(exp_1,\ldots,exp_n),b} P' \underset{\mathcal{L}}{\bowtie} Q} \quad a(exp_1,\ldots,exp_n) \notin \mathcal{L}$$

Choice Right

$$\frac{Q \xrightarrow{a(exp_1,\ldots,exp_n),b} Q'}{P + Q \xrightarrow{a(exp_1,\ldots,exp_n),b} Q'}$$

Coop Right

$$\frac{Q \xrightarrow{a(exp_1,\ldots,exp_n),b} Q'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{a(exp_1,\ldots,exp_n),b} P \underset{\mathcal{L}}{\bowtie} Q'} \quad a(exp_1,\ldots,exp_n) \notin \mathcal{L}$$

Synchronisation

$$\frac{P \xrightarrow{a(exp_1,\ldots,exp_n),b_1} P' \quad Q \xrightarrow{a(exp_1,\ldots,exp_n),b_2} Q'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{a(exp_1,\ldots,exp_n),b_1 \wedge b_2} P' \underset{\mathcal{L}}{\bowtie} Q'} \; a(exp_1,\ldots,exp_n) \in \mathcal{L}$$

IfThenElse True

$$\frac{P \xrightarrow{a(exp_1,\ldots,exp_n),b} P'}{[bexp]?P : Q \xrightarrow{a(exp_1,\ldots,exp_n),b \wedge bexp} P'}$$

IfThenElse False

$$\frac{Q \xrightarrow{a(exp_1,\ldots,exp_n),b} Q'}{[bexp]?P : Q \xrightarrow{a(exp_1,\ldots,exp_n),b \wedge \neg bexp} Q'}$$

Agent

$$\frac{P \xrightarrow{a(exp_1,\ldots,exp_n),b} P'}{A(k_1,\ldots,k_{n'}) \xrightarrow{a(z_1,\ldots,z_n),true} P''}$$

$A(i_1,\ldots,i_{n'}) \triangleq P \wedge\ \Gamma \vdash P' \to P''$
$\wedge\, \Gamma \vdash exp_1 \to z_1 \wedge \cdots \wedge \Gamma \vdash exp_n \to z_n$
$\wedge \Gamma \vdash b \to true \wedge \forall j.\ k_j, z_j \in \mathbb{R},$
with $\Gamma = \{(i_1,k_1),\ldots,(i_{n'},k_{n'})\}$

Figure 3.9: Semantics of parametric simple process algebra, part one of two.

Evaluate Exp Constant

$$\frac{}{\Gamma \vdash k \to k} \text{ if } k \in \mathbb{R}$$

Evaluate Variable

$$\frac{}{\Gamma \vdash i \to k} \text{ if } i \in Names \wedge \Gamma(i) = k$$

Evaluate Exp Binary Operator

$$\frac{\Gamma \vdash exp_1 \to k_1 \quad \Gamma \vdash exp_2 \to k_2}{\Gamma \vdash exp_1 \; op \; exp_2 \to k_3} \text{ if } k_3 = k_1 \; op \; k_2 \wedge op \in \{+, -, =, <\}$$

Evaluate Division

$$\frac{\Gamma \vdash exp_1 \to k_1}{\Gamma \vdash exp_1/k' \to k_2} \text{ if } k_2 = k_1/k'$$

Evaluate Bexp Constant

$$\frac{}{\Gamma \vdash b \to b} \text{ if } b \in \{true, false\}$$

Evaluate Bexp Binary Operator

$$\frac{\Gamma \vdash bexp_1 \to b_1 \quad \Gamma \vdash bexp_2 \to b_2}{\Gamma \vdash bexp_1 \; op \; bexp_2 \to b_3} \text{ if } b_3 = b_1 \; op \; b_2 \wedge op \in \{\wedge, \vee\}$$

Evaluate Negation

$$\frac{\Gamma \vdash bexp \to b}{\Gamma \vdash \neg bexp \to b'} \text{ if } b' = \neg b$$

Evaluate Process Constant

$$\frac{}{\Gamma \vdash nil \to nil}$$

Evaluate Prefix

$$\frac{\Gamma \vdash exp_1 \to k_1 \cdots \Gamma \vdash exp_n \to k_n \quad \Gamma \vdash P \to P'}{\Gamma \vdash a(exp_1, \ldots, exp_n).P \to a(k_1, \ldots, k_n).P'}$$

Evaluate Choice

$$\frac{\Gamma \vdash P \to P' \quad \Gamma \vdash Q \to Q'}{\Gamma \vdash P + Q \to P' + Q'}$$

Evaluate Agent

$$\frac{\Gamma \vdash exp_1 \to k_1 \cdots \Gamma \vdash exp_n \to k_n}{\Gamma \vdash A(exp_1, \ldots, exp_n) \to A(k_1, \ldots, k_n)}$$

Evaluate Cooperation

$$\frac{\Gamma \vdash P \to P' \quad \Gamma \vdash Q \to Q'}{\Gamma \vdash P \underset{\mathcal{L}}{\bowtie} Q \to P' \underset{\mathcal{L}'}{\bowtie} Q'} \; \mathcal{L}' = \{a_1(k_{11}, \ldots, k_{1n_1}), \ldots, a_m(k_{m1}, \ldots, k_{mn_m})\} \wedge \mathcal{L} = \{a_1(exp_{11}, \ldots, exp_{1n_1}), \ldots, a_m(exp_{m1}, \ldots, exp_{mn_m})\} \wedge \Gamma \vdash exp_{ij} \to k_{ij} \text{ for appropriate } i, j$$

Evaluate IfThenElse True

$$\frac{\Gamma \vdash bexp \to true \quad \Gamma \vdash P \to P'}{\Gamma \vdash [bexp]?P : Q \to P'}$$

Evaluate IfThenElse False

$$\frac{\Gamma \vdash bexp \to false \quad \Gamma \vdash Q \to Q'}{\Gamma \vdash [bexp]?P : Q \to Q'}$$

Figure 3.10: Semantics of parametric simple process algebra, part two of two. Operations on the right hand side of rules are evaluated.

- $a(exp_1, \ldots, exp_n)$ is a parametric action, $a(exp_1, \ldots, exp_n) \in Pactions$, with $Pactions$ the set of parametric actions and $exp_1, \ldots, exp_n$ a non empty list of expressions;
- $a(exp_1, \ldots, exp_n).P$ expresses the fact that parametric action $a(exp_1, \ldots, exp_n)$ has to be performed in order to change process $a(exp_1, \ldots, exp_n).P$ into the new process P;
- $P + P$ expresses the non deterministic choice between two processes. Once one is chosen, the other is discarded;
- $P \underset{\mathcal{L}}{\bowtie} P$ expresses the cooperation between two independent processes via the cooperation set $\mathcal{L}$, with $\mathcal{L} \subseteq Pactions$;
- $bexp$ is a boolean expression, defined as the constant $true$ or $false$, the equality test of expressions $exp_1 = exp_2$, the "less than" test $exp_1 < exp_2$, the conjunction or disjunction of two boolean expressions or the negation of a boolean expression;
- $[bexp]?P : Q$ corresponds to P if the evaluation of $bexp$ returns true or Q if it returns false;
- $A(exp_1, \ldots, exp_n)$ is used to recursively define processes, via the agent definition $A(i_1, \ldots, i_n) \triangleq P$, with $i_1, \ldots, i_n \in Names$ and $n \in (\mathbb{N} \setminus \{0\})$.

The semantics of pSPA is shown in Figures 3.9 and 3.10. The use of parameters may reduce significantly the length of the description of a process algebra model. In following two sections we illustrate the use of parameters in two examples. We will come back to parameters when we introduce the syntax used for the implementation of process algebra with hooks in Section 7.

3.3.1 Parametric Simple Process Algebra and Biochemistry

We now rewrite process definitions for the example in Section 3.1.1 using pSPA. In particular, we can use the notation $A(1, 2)$ to indicate the concentration of species **A** in location 1 is 2.

$$
\begin{aligned}
A(i,j) &\triangleq [0<j]?a(i).A(i,j-1) : nil \\
B(i,j) &\triangleq [0<j]?a(i).B(i,j-1) : nil \\
C(i,j) &\triangleq [0<j]? \\
&\quad ([j<2]? \\
&\qquad a(i).C(i,j+1) + t(i+1,i).C(i,j+1) \\
&\qquad +t(i,i+1).C(i,j-1) + t(i-1,i).C(i,j+1) \\
&\qquad +t(i,i-1).C(i,j-1) \\
&\qquad : t(i,i+1).C(i,j-1) + t(i,i-1).C(i,j-1)) \\
&\quad : a(i).C(i,j+1) + t(i+1,i).C(i,j+1) + t(i-1,i).C(i,j+1)
\end{aligned}
$$

The new initial state is:

$$
\begin{aligned}
&nil \underset{\mathcal{H}}{\between} (A(0,2) \underset{\mathcal{L}}{\between} B(0,2) \underset{\mathcal{L}}{\between} C(0,0)) \\
&\qquad \underset{\mathcal{K}}{\between} (A(1,2) \underset{\mathcal{L}'}{\between} B(1,2) \underset{\mathcal{L}'}{\between} C(1,0)) \underset{\mathcal{K}'}{\between} (A(2,0) \underset{\mathcal{L}''}{\between} B(2,0) \underset{\mathcal{L}''}{\between} C(2,2))
\end{aligned}
$$

where $\mathcal{H} = \{t(-1,0), t(0,-1), t(2,3), t(3,2)\}$ $\mathcal{L} = \{a(0)\}$, $\mathcal{L}' = \{a(1)\}$, $\mathcal{L}'' = \{a(2)\}$, $\mathcal{K} = \{t(0,1), t(1,0)\}$ and $\mathcal{K}' = \{t(1,2), t(2,1)\}$.

Examples of valid derivations are:

$$
\begin{aligned}
&nil \underset{\mathcal{H}}{\between} (A(0,2) \underset{\mathcal{L}}{\between} B(0,2) \underset{\mathcal{L}}{\between} C(0,0)) \\
&\qquad \underset{\mathcal{K}}{\between} (A(1,2) \underset{\mathcal{L}'}{\between} B(1,2) \underset{\mathcal{L}'}{\between} C(1,0)) \underset{\mathcal{K}'}{\between} (A(2,0) \underset{\mathcal{L}''}{\between} B(2,0) \underset{\mathcal{L}''}{\between} C(2,2)) \\
&\xrightarrow{a(1),true} nil \underset{\mathcal{H}}{\between} (A(0,2) \underset{\mathcal{L}}{\between} B(0,2) \underset{\mathcal{L}}{\between} C(0,0)) \\
&\qquad \underset{\mathcal{K}}{\between} (A(1,1) \underset{\mathcal{L}'}{\between} B(1,1) \underset{\mathcal{L}'}{\between} C(1,1)) \underset{\mathcal{K}'}{\between} (A(2,0) \underset{\mathcal{L}''}{\between} B(2,0) \underset{\mathcal{L}''}{\between} C(2,2))
\end{aligned}
$$

and

$$
\begin{aligned}
&nil \underset{\mathcal{H}}{\between} (A(0,2) \underset{\mathcal{L}}{\between} B(0,2) \underset{\mathcal{L}}{\between} C(0,0)) \\
&\qquad \underset{\mathcal{K}}{\between} (A(1,2) \underset{\mathcal{L}'}{\between} B(1,2) \underset{\mathcal{L}'}{\between} C(1,0)) \underset{\mathcal{K}'}{\between} (A(2,0) \underset{\mathcal{L}''}{\between} B(2,0) \underset{\mathcal{L}''}{\between} C(2,2)) \\
&\xrightarrow{t(2,1),true} nil \underset{\mathcal{H}}{\between} (A(0,2) \underset{\mathcal{L}}{\between} B(0,2) \underset{\mathcal{L}}{\between} C(0,0)) \\
&\qquad \underset{\mathcal{K}}{\between} (A(1,2) \underset{\mathcal{L}'}{\between} B(1,2) \underset{\mathcal{L}'}{\between} C(1,1)) \underset{\mathcal{K}'}{\between} (A(2,0) \underset{\mathcal{L}''}{\between} B(2,0) \underset{\mathcal{L}''}{\between} C(2,1))
\end{aligned}
$$

3.3.2 Parametric Simple Process Algebra and Tissue Growth

Consider the model of tissue growth in Section 3.1.2. We extend our previous approach using pSPA.

Explicit Modelling of Empty Space. Using pSPA we can define generic *Empty* and *Tissue* processes as follows:

$$\begin{aligned} &Empty(i) \triangleq mito(i-1,i).Tissue(i) + mito(i+1,i).Tissue(i) \\ &Tissue(i) \triangleq apo(i).Empty(i) + mito(i,i-1).Tissue(i) \\ &+mito(i,i+1).Tissue(i) \end{aligned}$$

The initial state of the model is given by the following process:

$$nil \underset{\mathcal{L}}{\bowtie} Empty(0) \underset{\mathcal{K}}{\bowtie} Tissue(1) \underset{\mathcal{H}}{\bowtie} Empty(2)$$

where $\mathcal{L} = \{mito(-1,0), mito(0,-1), mito(3,2), mito(2,3)\}$, $\mathcal{K} = \{mito(0,1), mito(1,0)\}$ and $\mathcal{H} = \{mito(1,2), mito(2,1)\}$. Here we use $nil \underset{\mathcal{L}}{\bowtie}$ to specify the boundaries of the model.

Examples of valid derivations are:

$$\begin{aligned} &nil \underset{\mathcal{L}}{\bowtie} Empty(0) \underset{\mathcal{K}}{\bowtie} Tissue(1) \underset{\mathcal{H}}{\bowtie} Empty(2) \\ &\xrightarrow{mito(1,2)} nil \underset{\mathcal{L}}{\bowtie} Empty(0) \underset{\mathcal{K}}{\bowtie} Tissue(1) \underset{\mathcal{H}}{\bowtie} Tissue(2) \end{aligned}$$

and

$$\begin{aligned} &nil \underset{\mathcal{L}}{\bowtie} Empty(0) \underset{\mathcal{K}}{\bowtie} Tissue(1) \underset{\mathcal{H}}{\bowtie} Empty(2) \\ &\xrightarrow{apo(1)} nil \underset{\mathcal{L}}{\bowtie} Empty(0) \underset{\mathcal{K}}{\bowtie} Empty(1) \underset{\mathcal{H}}{\bowtie} Empty(2) \end{aligned}$$

3.4 Summary

In this chapter we have illustrated how biological entities and events can be represented by processes and actions using a simple process algebra with multi-way synchronisation. Then we extended the algebra with a stochastic semantics which permits the use of functional rates for the actions. Concepts of *open* and *closed* activities have been introduced to determine whether a functional rate can be evaluated or not. These concepts are inspired by the observation that biological interactions require a set of participants that can be determined beforehand.

Finally, we defined a parametrised version of the simple process algebra, to reduce the length of model definitions. Several examples using simple process algebra and its extensions have been given throughout the chapter.

In the next chapter we address the problem of modelling multiple spatial scales and the interactions within and between spatial scales.

Chapter 4

Multi-Scale Modelling with Process Algebra with Priorities

In this chapter we discuss mechanisms of interactions between spatial scales of biological systems. After an introduction of possible relationships between scales at the beginning of Section 4.1, we focus on modelling dependencies between the concentration level of biochemical species at the molecular scale and the behaviour of the cell at the cellular scale in Section 4.1.1. In the same section we demonstrate that the simple process algebra defined in Chapter 3 is not effective in modelling such dependencies.

In Section 4.2 we investigate the use of a process algebra with action priorities to model the interactions between scales in a more effective way. We show it is suited to the task, as we highlight in Sections 4.2.1, 4.2.2 and 4.2.3. However, this algebra has not been designed to model multi-scale scenarios and fails to address in its syntax and semantics some of the issues that arise from this context. We conclude with a discussion of drawbacks of the use of process algebra with priorities in Section 4.2.4. For completeness we also provide a stochastic semantics based on functional rates in Appendix A.

4.1 Mechanisms of Interaction Between Scales

In this thesis we are primarily concerned with spatial scales and we assume that events take place at the same time scale.

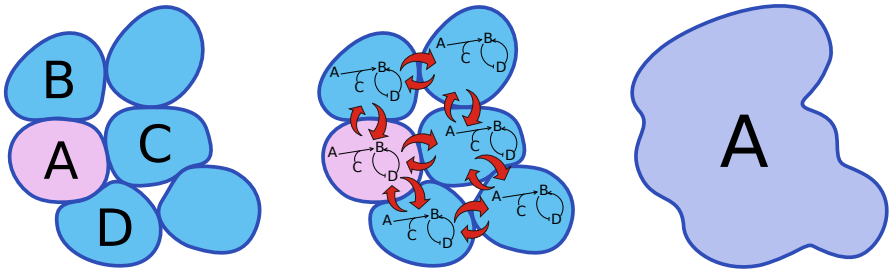


Figure 4.1: Tissue infection at different abstraction levels: on the left, cellular scale; centre: molecular scale; on the right: tissue scale.

In this setting we want to model relationships between scales such as:

- **Abstractions.** Entities and events at a given scale can be described in more detail using the entities and events of another scale. For example, cells are composed of molecules and cell movement is the result of molecular interactions.
- **Interactions.** The behaviour of entities at a given scale changes depending on events performed at other scales. For example, if a cell finds nutrients, molecular digestive processes can be performed.

An example of abstraction is illustrated in Figure 4.1. At the cellular scale (on the left of the figure), cell **A** is infected and can infect cells **B**, **C** or **D**. At the molecular scale (centre), molecules can move between cells and a specific molecular configuration (i.e. concentration of each species) can be associated with the cellular phenotype of cell infection. At the tissue scale (on the right), a portion of tissue presents a certain degree of infection, determined by the number of infected cells.

An example of interaction is illustrated in Figure 4.2. A dependency is defined between the molecular scale (on the left of the figure) and the cellular scale (on the right of the figure): cellular duplication is possible if and only if the concentration of molecule **A** is above a certain threshold. In other words, high concentration of molecule **A** activates the ability of the cell to duplicate.

Other examples of dependencies are illustrated in Figure 4.3. If a cell dies (top of figure) this implies the concentration of the molecules inside it is dispersed. If a cell **C** duplicates (bottom of figure), independent concentrations of molecules originally in **C** will be present in both the resulting cells **C**′ and **C**″.

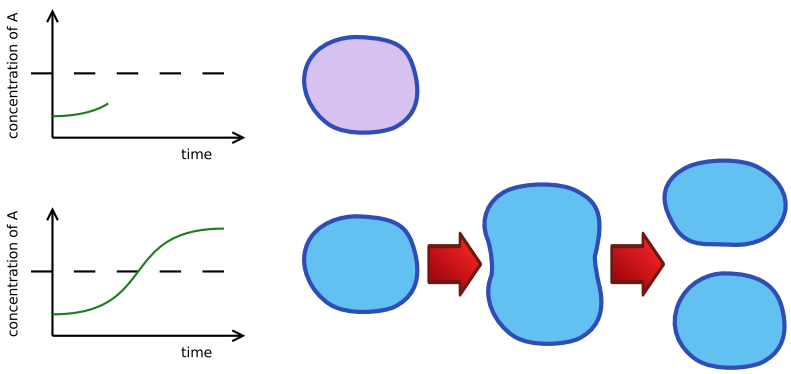


Figure 4.2: Interactions between scales. Only if the concentration of a certain molecule (molecular scale) is high, then a cell can duplicate (cellular scale).

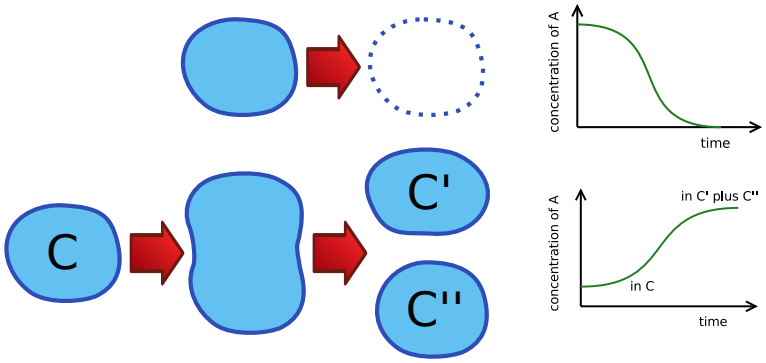


Figure 4.3: Dependencies between scales. Cellular events such as death and duplication (cellular scale), imply changes in concentration of molecules (molecular scale) inside the cells.

In the next section we attempt to implement dependencies between scales with the simple process algebra defined in Chapter 3.

4.1.1 Modelling Thresholds with Simple Process Algebra

Dependencies between scales can be modelled with action synchronisations between processes that represent different scales. This can be implemented easily using SPA as long as actions at a given scale always have the same effect at other scales. However, it is possible that entities at a certain scale perform the same action many times and only *some* instances of that action affect other scales. Usually this involves some mechanism of *memory*, or *counting*, in order to know *when* an action at a scale affects other scales. An example is the dependency

$A_3 \triangleq a.A_2 \quad B_3 \triangleq nil$

$A_2 \triangleq a.A_1 \quad B_2 \triangleq a.B_3$

$A_1 \triangleq a.A_0 \quad B_1 \triangleq a.B_2$

$A_0 \triangleq nil \quad B_0 \triangleq a.B_1$

$C_1 \triangleq nil$

$C_0 \triangleq x.C_1$

Initial state: $(A_3 \underset{\{a\}}{\bowtie} B_0) \underset{\{x\}}{\bowtie} C_0$

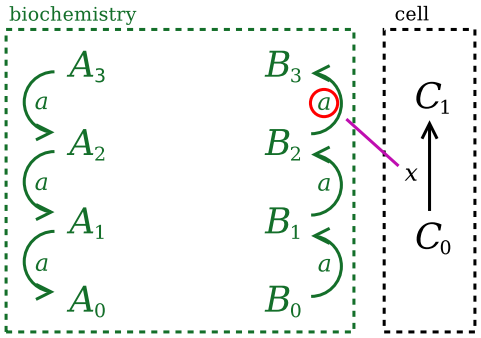


Figure 4.4: Example of a threshold definition problem in simple process algebra.

between the behaviour of a cell and the concentration of molecules inside the cell, as depicted in Figure 4.2. In this case, biochemical reactions are performed continuously inside a cell, but only *some* instances of these reactions are responsible for the change in behaviour of the cell (because a threshold has been reached).

An example of the problem of defining a dependency between a concentration threshold and the behaviour of a cell in SPA is illustrated in Figure 4.4. SPA processes are on the left while their graphical representation is on the right of the figure. Action a represents a biochemical reaction that converts concentration of molecule **A** into concentration of molecule **B**. Action x represents a change in the behaviour of the cell, from C_0 to C_1. The circled a action indicates that a concentration threshold for **B** is crossed. If we assume that the behaviour of the cell associated with high **B** concentration is C_1, then x should be performed in synchronisation with the circled a action, or at least performed immediately after it, with no delay.

With the following three examples we illustrate the limitations of SPA in dealing with this particular problem.

Example 1. Consider the following example. In a cell there are two molecular species **A** and **B**, involved in the following reactions:

$$R_a : \rightarrow \mathbf{A} \qquad R_b : \mathbf{A} \rightarrow$$

$$R_c : \rightarrow \mathbf{B} \qquad R_d : \mathbf{B} \rightarrow$$

$$R_e : \mathbf{A} \rightarrow \mathbf{B} \qquad R_f : \mathbf{B} \rightarrow \mathbf{A}$$

We consider a maximum number of levels of concentration equal to 2. The processes for **A** and **B** are:

$$A_0 \triangleq a.A_1 + f.A_1 \quad A_1 \triangleq a.A_2 + f.A_2 + b.A_0 + e.A_0 \quad A_2 \triangleq b.A_1 + e.A_1$$
$$B_0 \triangleq c.B_1 + e.B_1 \quad B_1 \triangleq c.B_2 + e.B_2 + d.B_0 + f.B_0 \quad B_2 \triangleq d.B_1 + f.B_1$$

In addition, the cell that contains species **A** and **B** can either move or absorb nutrients. This is represented by the following process:

$$Cell \triangleq move.Cell + absorb.Cell$$

The initial state is:

$$(A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\emptyset}{\bowtie} Cell$$

In the proposed model, the molecular scale and the cellular scale are independent. The cell can move and absorb nutrients regardless of what biochemical reactions take place.

Example 2. Consider now the introduction of a dependency between scales. Assume that the cell absorbs nutrients and does not move as long as the concentration level of molecule **B** is high, i.e. 2. If the concentration level of **B** is not 2, then the cell moves continuously, ignoring nutrients. In order to model this dependency, process *Cell* needs to synchronise with processes B_1 and B_2. This requires the use of new actions to represent crossing concentration threshold of **B**, from 1 to 2. Moreover, processes A_0, A_1 and A_2 need to be updated so they can still synchronise with **B** to perform reactions R_e and R_f. The updated model is the following:

$$\begin{array}{lll} A_0 \triangleq a.A_1 + f.A_1 & A_1 \triangleq a.A_2 + f.A_2 + b.A_0 + e.A_0 & A_2 \triangleq b.A_1 + e.A_1 \\ \quad + fy.A_1 & \quad + ex.A_0 + fy.A_2 & \quad + ex.A_1 \\ B_0 \triangleq c.B_1 + e.B_1 & B_1 \triangleq cx.B_2 + ex.B_2 + d.B_0 + f.B_0 & B_2 \triangleq dy.B_1 + fy.B_1 \end{array}$$

$$Cell_M \triangleq move.Cell_M + cx.Cell_A + ex.Cell_A$$
$$Cell_A \triangleq absorb.Cell_A + dy.Cell_M + fy.Cell_M$$

The new initial state is:

$$(A_1 \underset{\{e,ex,f,fy\}}{\bowtie} B_1) \underset{\{cx,dy,ex,fy\}}{\bowtie} Cell_M$$

The dependency between the concentration of **B** and the behaviour of the cell has been introduced with the addition of actions cx, dy, ex and fy. These actions

represent a threshold crossing, with suffix x denoting that the concentration of **B** changes from level 1 to 2 and suffix y denoting that the concentration of **B** changes from 2 to 1. Most importantly, as a consequence of the introduction of these new actions, the processes that represent **A** must be altered, in order to synchronise with the new actions ex and fy.

Example 3. Consider now a more complex dependency between the molecular and the cellular scale. Assume that the cell absorbs nutrients if and only if a specific configuration of the biochemistry is present. In particular, if the concentrations of **A** and **B** are high at the same time, i.e. both level 2, then the cell absorbs nutrients; it moves otherwise. This new dependency requires the definition of two thresholds and the modification of the processes representing the cell, the latter to monitor how many species have passed the threshold.

The updated model is as follows:

$$\begin{array}{lll} A_0 \triangleq a.A_1 + f.A_1 & A_1 \triangleq ax.A_2 + fx.A_2 + b.A_0 + e.A_0 & A_2 \triangleq by.A_1 + ey.A_1 \\ \quad +fy.A_1 & \quad +ex.A_0 + fxy.A_2 & \quad +exy.A_1 \\ B_0 \triangleq c.B_1 + e.B_1 & B_1 \triangleq cx.B_2 + ex.B_2 + d.B_0 + f.B_0 & B_2 \triangleq dy.B_1 + fy.B_1 \\ \quad +ey.B_1 & \quad +fy.B_0 + exy.B_2 & \quad +fxy.B_1 \end{array}$$

$$\begin{aligned} Cell_{M0} &\triangleq move.Cell_{M0} + ax.Cell_{M1} + cx.Cell_{M1} + ex.Cell_{M1} + fx.Cell_{M1} \\ Cell_{M1} &\triangleq move.Cell_{M1} + ax.Cell_A + cx.Cell_A + ex.Cell_A + fx.Cell_A \\ &\quad +by.Cell_{M0} + dy.Cell_{M0} + ey.Cell_{M0} + fy.Cell_{M0} \\ &\quad +exy.Cell_{M1} + fxy.Cell_{M1} \\ Cell_A &\triangleq absorb.Cell_A + by.Cell_{M1} + dy.Cell_{M1} + ey.Cell_{M1} + fy.Cell_{M1} \end{aligned}$$

The new initial state is:

$$(A_1 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{K}}{\bowtie} Cell_{M0}$$

where $\mathcal{L} = \{e,\ ex,\ ey,\ exy,\ f,\ fx,\ fy,\ fxy\}$ and $\mathcal{K} = \{ax,\ by,\ cx,\ dy,\ ex,\ ey,\ exy,\ fx,\ fy,\ fxy\}$. The model definition is longer and more complex because cell processes need to synchronise with all actions representing the crossing of a threshold or the crossing of more than one threshold at the same time. Moreover, three processes are used to count how many species have crossed their threshold: $Cell_{M0}$, none of them, $Cell_{M1}$, one of them, and $Cell_A$, both of them.

It is evident that the addition of new thresholds increases the number of action names. In general, the definition of dependencies that require this or any similar type of counting may result in long descriptions that are difficult to read.

As a solution to this problem we propose to employ a process algebra with action priorities instead of SPA. In the next section we investigate this approach.

4.2 Process Algebra with Priorities

The syntax for a process algebra with multi-way synchronisation and action priorities (PAwP) is:

$$P ::= nil \mid p{:}a.P \mid P + P \mid P \underset{\mathcal{L}}{\bowtie} P \mid A$$

The only addition with respect to SPA introduced in Section 3.1, is the association of a priority $p \in (\mathbb{N} \setminus \{0\})$ to actions. We use $p{:}a$ to indicate that action a is performed with priority p. Moreover, we assume that for any two prioritised actions $p_1{:}x_1$ and $p_2{:}x_2$, if $x_1 = x_2$ then $p_1 = p_2$.

The main difference between the semantics of PAwP and the semantics of SPA defined in Section 3.1 is that we have to specify how priorities are handled. In general we want to execute only the actions that have the highest priority. As explained in (Bernardo, 1996), only if we know all the potential moves from a state, we can correctly select the actions with the highest priority.

We now introduce a semantics for PAwP, based on the semantics of *extended Markovian process algebra* (EMPA) (Bernardo, 1996). Note that EMPA has a stochastic semantics and here we adopt only its treatment of priorities. With $\{\!|\}$ delimiting a multi set and $\uplus$ the union of multi sets, we use the following definitions:

- $PM(P)$ returns the potential moves of a process P, $PM(P) \subseteq (\mathbb{N} \setminus \{0\}) \times \mathit{Actions} \times \mathbb{P}$. $PM(P)$ is defined by structural induction as:

  $PM(nil) = \emptyset$

  $PM(p{:}a.P) = \{\!|(p, a, P)|\!\}$

  $PM(P_1 + P_2) = PM(P_1) \uplus PM(P_2)$

  $$\begin{aligned} PM(P_1 \underset{\mathcal{L}}{\bowtie} P_2) = \{\!|(p, a, P_1' \underset{\mathcal{L}}{\bowtie} P_2) \mid (p, a, P_1') \in PM(P_1) \wedge a \notin \mathcal{L}\} \\ \uplus \{(p, a, P_1 \underset{\mathcal{L}}{\bowtie} P_2') \mid (p, a, P_2') \in PM(P_2) \wedge a \notin \mathcal{L}|\!\} \\ \uplus \{\!|(p, a, P_1' \underset{\mathcal{L}}{\bowtie} P_2') \mid (p, a, P_1') \in PM(P_1) \wedge (p, a, P_2') \\ \in PM(P_2) \ \wedge a \in \mathcal{L}|\!\} \end{aligned}$$

  $PM(A) = PM(P)$ if $A \triangleq P$

- $Select(PM(P))$ returns the potential moves of P with the highest priority and is defined as:

$$Select(PMSet) = \{\!|(p,a,P) \mid (p,a,P) \in PMSet \wedge \forall (q,b,Q) \in PMSet, \\ p \geq q|\!\}$$

Using the functions defined above, the semantics of PAwP is given by the following derivation rule:

$$\frac{(p,a,P') \in Select(PM(P))}{P \xrightarrow{a} P'}$$

In the next section we show how the threshold examples in Section 4.1.1 can be modelled more effectively with PAwP.

4.2.1 Modelling Thresholds with Process Algebra with Priorities

Consider Examples 1, 2 and 3 in Section 4.1.1. We now illustrate how these examples can be modelled with PAwP. The idea is that actions with priority 1 represent events that happen *within* a scale, while actions with priority higher than 1 are interrupts that occur *between* scales.

Example 1 presents two independent scales and can be converted from simple process algebra to process algebra with priorities by simply assigning priority 1 to all actions.

Example 2 presents a dependency between scales: a cell absorbs nutrients if concentration level of molecule **B** is equal to 2, i.e. it is high, or moves if the concentration level of **B** is either 0 or 1, i.e. is not high. Model definition in process algebra with priorities is as follows:

$$A_0 \triangleq 1{:}a.A_1 + 1{:}f.A_1 \quad A_1 \triangleq 1{:}a.A_2 + 1{:}f.A_2 + 1{:}b.A_0 + 1{:}e.A_0 \quad A_2 \triangleq 1{:}b.A_1 + 1{:}e.A_1$$

$$B_0 \triangleq 1{:}c.B_1 + 1{:}e.B_1 \quad B_1 \triangleq 1{:}c.2{:}x.B_2 + 1{:}e.2{:}x.B_2 + 1{:}d.B_0 + 1{:}f.B_0 \quad B_2 \triangleq 1{:}d.2{:}y.B_1 + 1{:}f.2{:}y.B_1$$

$$Cell_M \triangleq 1{:}move.Cell_M + 2{:}x.Cell_A$$

$$Cell_A \triangleq 1{:}absorb.Cell_A + 2{:}y.Cell_M$$

The new initial state is:

$$(A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M$$

Example of valid transitions are:

$$(A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M \xrightarrow{b} (A_0 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M$$

$$(A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M \xrightarrow{move} (A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M$$

$$(A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M \xrightarrow{e} (A_0 \underset{\{e,f\}}{\bowtie} 2{:}x.B_2) \underset{\{x,y\}}{\bowtie} Cell_M$$

Most importantly, from state $(A_0 \underset{\{e,f\}}{\bowtie} 2{:}x.B_2) \underset{\{x,y\}}{\bowtie} Cell_M$ the only derivation possible is:

$$(A_0 \underset{\{e,f\}}{\bowtie} 2{:}x.B_2) \underset{\{x,y\}}{\bowtie} Cell_M \xrightarrow{x} (A_0 \underset{\{e,f\}}{\bowtie} B_2) \underset{\{x,y\}}{\bowtie} Cell_A$$

Compare the above definition with Example 1 of Section 4.1.1. The only additions are actions with priority 2 (x and y), placed right after an action crosses the concentration threshold for molecule **B**. These actions are used as *interrupts* to communicate a change in cell behaviour to the processes representing the cell.

Compare now the definition with Example 2 of Section 4.1.1. The following improvements can be noted:

- only the processes representing the concentration of **B** and the behaviour of the cell have been altered, while processes representing the behaviour of **A** are left unaltered;
- the cell processes do not synchronise with as many specific action names from the molecular scale as in Example 2, but only with action names x and y, which just indicate threshold crossing;
- in this particular example we have a clear distinction between actions that happen *within* a scale (priority 1) and actions that happen *between* scales (priority 2).

In conclusion, the description is more compact and more readable.

In the next section we propose a PAwP model of Example 3 in Section 4.1.1. After that, in Section 4.2.3 we propose a multi-scale model of tissue growth.

4.2.2 Process Algebra with Priorities and a Three Layers Example

In this section we introduce an example similar to Example 3 in Section 4.1.1. With this example we show how PAwP can be used to model the flow of inter-scale information between three scales. The result is a more readable model with improved compositionality.

Recall the scenario of Example 3 in Section 4.1.1. A cell absorbs nutrients and does not move when the concentration of both molecules **A** and **B** is high, while the same cell moves and does not absorb nutrients when at least one of **A** and **B** is not present in high concentration. The reactions performed at the molecular scale are:

$$R_a : \rightarrow \mathbf{A} \qquad R_b : \mathbf{A} \rightarrow$$
$$R_c : \rightarrow \mathbf{B} \qquad R_d : \mathbf{B} \rightarrow$$
$$R_e : \mathbf{A} \rightarrow \mathbf{B} \qquad R_f : \mathbf{B} \rightarrow \mathbf{A}$$

In this implementation we use additional processes to count how many of species **A** and **B** present high concentration. We use P_0 to indicate none of them, P_1 one of them and P_2 both of them. These intermediate processes will then communicate to the processes representing the cell when a change in cell behaviour has to take place. This is in contrast with our implementation of Example 3 in Section 4.1.1, where the role of counting was assigned directly to the processes representing the cell. This different approach demonstrates that more than two scales can be used. Moreover, the use of P_0 as an intermediate layer between the molecular and cellular scale improves the compositionality of the model: if one desires to test other dependencies between scales, then process P_0 can be substituted with a different process to count, for example, if *at least* one of **A** and **B** is above a concentration threshold or if exactly one of the two is.

The PAwP model is defined by the following processes:

$$A_0 \triangleq 1{:}a.A_1 + 1{:}f.A_1 \qquad B_0 \triangleq 1{:}c.B_1 + 1{:}e.B_1 \qquad P_0 \triangleq 2{:}p.P_1$$

$$A_1 \triangleq 1{:}a.2{:}p.A_2 + 1{:}b.A_0 + 1{:}f.2{:}p.A_2 + 1{:}e.A_0 \qquad B_1 \triangleq 1{:}c.2{:}p.B_2 + 1{:}d.B_0 + 1{:}e.2{:}p.B_2 + 1{:}f.B_0 \qquad P_1 \triangleq 2{:}q.P_0 + 2{:}p.3{:}x.P_2$$

$$A_2 \triangleq 1{:}b.2{:}q.A_1 + 1{:}e.2{:}q.A_1 \qquad B_2 \triangleq 1{:}d.2{:}q.B_1 + 1{:}f.2{:}q.B_1 \qquad P_2 \triangleq 2{:}q.3{:}y.P_1$$

$$Cell_M \triangleq 3{:}x.Cell_A + 1{:}move.Cell_M \qquad Cell_A \triangleq 3{:}y.Cell_M + 1{:}absorb.Cell_A$$

The initial state of the model is given by:

$$((A_0 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$$

where $\mathcal{L} = \{e, f\}$, $\mathcal{H} = \{p, q\}$ and $\mathcal{K} = \{x, y\}$. The states of the model, reachable from the initial state, are:

1: $((A_0 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$
2: $((A_0 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$
3: $((A_1 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$
4: $((A_1 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$
5: $((A_0 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
6: $((A_1 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
7: $((A_2 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
8: $((A_2 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
9: $((A_2 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} P_2) \underset{\mathcal{K}}{\bowtie} Cell_A$
10: $((A_0 \underset{\mathcal{L}}{\bowtie} 2{:}p.B_2) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$
11: $((A_0 \underset{\mathcal{L}}{\bowtie} 2{:}q.B_1) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
12: $((2{:}p.A_2 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$
13: $((2{:}q.A_1 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
14: $((A_1 \underset{\mathcal{L}}{\bowtie} 2{:}q.B_1) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
15: $((2{:}q.A_1 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
16: $((A_1 \underset{\mathcal{L}}{\bowtie} 2{:}p.B_2) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$
17: $((2{:}p.A_2 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$
18: $((A_2 \underset{\mathcal{L}}{\bowtie} 2{:}p.B_2) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
19: $((2{:}p.A_2 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
20: $((A_2 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} 3{:}x.P_2) \underset{\mathcal{K}}{\bowtie} Cell_M$
21: $((A_2 \underset{\mathcal{L}}{\bowtie} 2{:}q.B_1) \underset{\mathcal{H}}{\bowtie} P_2) \underset{\mathcal{K}}{\bowtie} Cell_A$
22: $((2{:}q.A_1 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} P_2) \underset{\mathcal{K}}{\bowtie} Cell_A$
23: $((A_2 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} 3{:}y.P_1) \underset{\mathcal{K}}{\bowtie} Cell_A$
24: $((A_1 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} 3{:}y.P_1) \underset{\mathcal{K}}{\bowtie} Cell_A$
25: $((2{:}q.A_1 \underset{\mathcal{L}}{\bowtie} 2{:}p.B_2) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
26: $((2{:}p.A_2 \underset{\mathcal{L}}{\bowtie} 2{:}q.B_1) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M$
27: $((2{:}q.A_1 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} 3{:}x.P_2) \underset{\mathcal{K}}{\bowtie} Cell_M$
28: $((A_2 \underset{\mathcal{L}}{\bowtie} 2{:}q.B_1) \underset{\mathcal{H}}{\bowtie} 3{:}x.P_2) \underset{\mathcal{K}}{\bowtie} Cell_M$

States and transitions are shown in Figure 4.5.

To illustrate the use of compositionality in this model, consider the change of the condition for the cell to absorb nutrients from "both **A** and **B** above the threshold" to "at least one of **A** and **B** above the threshold". To apply this change, it is sufficient to replace process P_0 with the following process Q_0:

$$Q_0 \triangleq 2{:}p.3{:}x.Q_1 \quad Q_1 \triangleq 2{:}q.3{:}y.Q_0 + 2{:}p.Q_2 \quad Q_2 \triangleq 2{:}q.Q_1$$

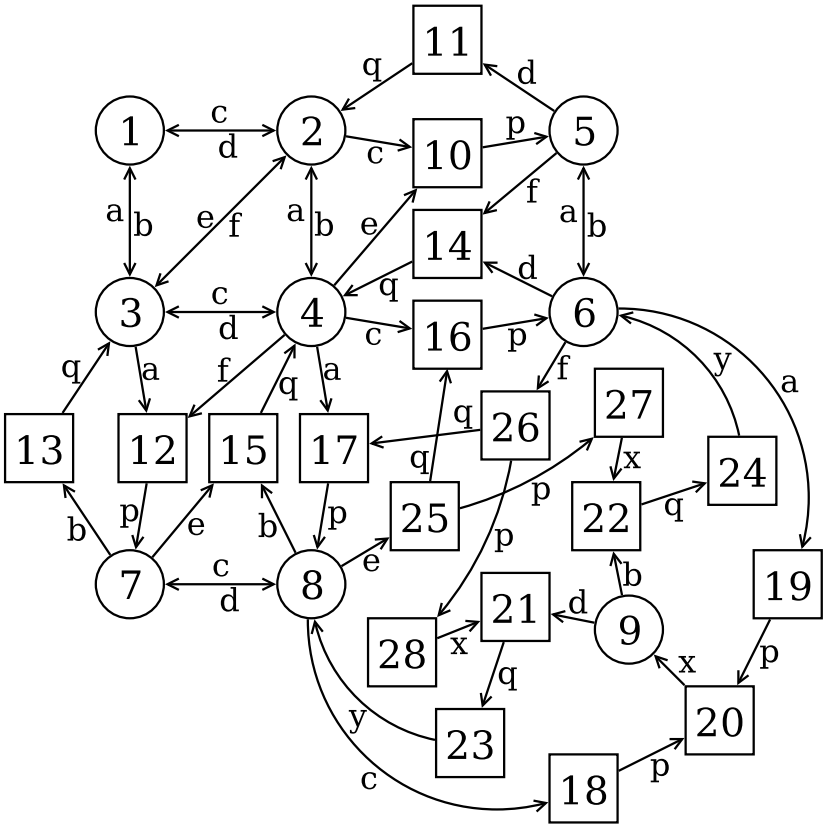


Figure 4.5: Reachable states and transitions generated by the example in Section 4.2.2. Squares are intermediate states, i.e. states that precede transitions labelled by actions with priority higher than 1. Auto transitions (actions *move* or *absorb* in states from 1 to 9) are not shown.

4.2.3 Process Algebra with Priorities and Tissue Growth with Biochemistry

In this section we propose an extension of the tissue growth example with explicit modelling of empty space introduced in Section 3.1.2. Here we combine the tissue scale with a biochemical scale and define dependencies between the two scales, producing a multi-scale model.

In analogy with the model proposed in Section 3.1.2, we consider three regions of space, R0, R1 and R2 which can be empty or can contain tissue. Here we consider two different types of tissue: active and inactive. *Active* tissue is tissue that can either die, or can grow (actions beginning with *mito*, for mitosis, cell duplication). *Inactive* tissue can only die and not grow. Empty regions are modelled with processes beginning with *Empty*, e.g. *Empty*1 if region R1 is

empty. Active tissue is modelled with processes beginning with *Tissueon*, e.g. *Tissueon*1, that can perform actions beginning with *apo*, for apoptosis, or actions beginning with *mito*, for mitosis, e.g. *mito*12 if growth happens from region R1 to region R2. Inactive tissue is modelled with processes beginning with *Tissueoff*, e.g. *Tissueoff*1, that can perform actions beginning with *apo*.

The biochemical scale consists of molecular species **A**, modelled using the processes as levels of concentration abstraction, with concentration levels from 0 to 2. Each region contains an independent concentration of **A**. Processes representing the concentration of **A** participate in biochemical reactions, here modelled with actions a and b, that respectively increment and decrease the concentration levels of **A** by one. In addition, we use processes prefixed with NA, that cannot perform any action, when the corresponding region is empty. For example, we use $NA1$ when R1 is empty and no biochemical reactions are possible.

Dependencies between the biochemical and the tissue scale are as follows:

- in a region, tissue is active if and only if the concentration level of **A** is 2, while the tissue is inactive otherwise. This dependency is modelled using the threshold system explained in Section 4.2.1. Actions beginning with *mitoon* and *mitooff* have priority 2 and are used to communicate to the tissue scale when a threshold has been crossed;

- no biochemical action takes place in empty space. This dependency is modelled by imposing that whenever a tissue process performs an apoptosis action, the process representing the concentration of **A** in the same region synchronises with that action, changing to an NA process. Analogously, an NA process changes to a concentration process whenever empty space turns into tissue.

The model is defined as follows:

$$NA0 \triangleq 1{:}mito10.A0_0$$
$$A0_0 \triangleq 1{:}apo0.NA0 + 1{:}a0.A0_1$$
$$A0_1 \triangleq 1{:}apo0.NA0 + 1{:}a0.2{:}mitoon0.A0_2 + 1{:}b0.A0_0$$
$$A0_2 \triangleq 1{:}apo0.NA0 + 1{:}b0.2{:}mitooff0.A0_1$$

$$Empty0 \triangleq 1{:}mito10.Tissueoff0$$
$$Tissueoff0 \triangleq 1{:}apo0.Empty0 + 2{:}mitoon0.Tissueon0$$
$$Tissueon0 \triangleq 1{:}apo0.Empty0 + 1{:}mito01.Tissueon0$$
$$+2{:}mitooff0.Tissueoff0$$

$$NA1 \triangleq 1{:}mito01.A1_0 + 1{:}mito21.A1_0$$
$$A1_0 \triangleq 1{:}apo1.NA1 + 1{:}a1.A1_1$$
$$A1_1 \triangleq 1{:}apo1.NA1 + 1{:}a1.2{:}mitoon1.A1_2 + 1{:}b1.A1_0$$
$$A1_2 \triangleq 1{:}apo1.NA1 + 1{:}b1.2{:}mitooff1.A1_1$$

$$Empty1 \triangleq 1{:}mito21.Tissueoff1 + 1{:}mito01.Tissueoff1$$
$$Tissueoff1 \triangleq 1{:}apo1.Empty1 + 2{:}mitoon1.Tissueon1$$
$$Tissueon1 \triangleq 1{:}apo1.Empty1 + 1{:}mito12.Tissueon1$$
$$+1{:}mito10.Tissueon1 + 2{:}mitooff1.Tissueoff1$$

$$NA2 \triangleq 1{:}mito12.A2_0$$
$$A2_0 \triangleq 1{:}apo2.NA2 + 1{:}a2.A2_1$$
$$A2_1 \triangleq 1{:}apo2.NA2 + 1{:}a2.2{:}mitoon2.A2_2 + 1{:}b2.A2_0$$
$$A2_2 \triangleq 1{:}apo2.NA2 + 1{:}b2.2{:}mitooff2.A2_1$$

$$Empty2 \triangleq 1{:}mito12.Tissueoff2$$
$$Tissueoff2 \triangleq 1{:}apo2.Empty2 + 2{:}mitoon2.Tissueon2$$
$$Tissueon2 \triangleq 1{:}apo2Empty2 + 1{:}mito21.Tissueon2$$
$$+2{:}mitooff2.Tissueoff2$$

The initial state of the model is defined by the following process:

$$(NA0 \underset{\emptyset}{\between} A1_1 \underset{\emptyset}{\between} A2_2) \underset{\mathcal{L}}{\between} (Empty0 \underset{\mathcal{K}}{\between} Tissueoff1 \underset{\mathcal{K}'}{\between} Tissueon2)$$

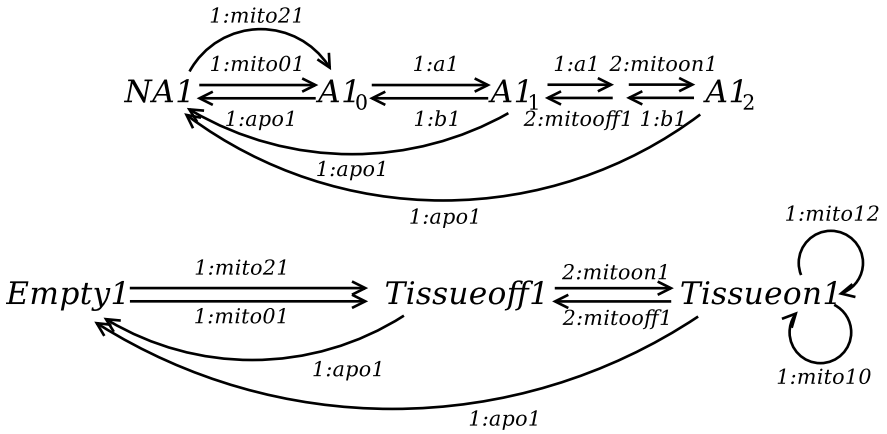


Figure 4.6: Graphical representation of processes $Empty1$ and $NA1$.

with $\mathcal{L} = \{mitoon0,\ mitooff0,\ mitoon1,\ mitooff1, mitoon2, mitooff2,\ apo0,\ apo1,\ apo2,\ mito10,\ mito01,\ mito21,\ mito12\}$, $\mathcal{K} = \{mito10, mito01\}$, $\mathcal{K} = \{mito21, mito12\}$. A graphical representation of processes $NA1$ and $Empty1$ is illustrated in Figure 4.6.

Examples of valid transitions are:

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$
$$\xrightarrow{b2} (NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} 2{:}mitooff2.A2_1) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$
$$\xrightarrow{apo1} (NA0 \underset{\emptyset}{\bowtie} NA1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Empty1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$

Notice that in this case there is only one biochemical species, **A**, and that the use of actions with priority higher than 2 is not necessary. In fact one could model the threshold as in Example 2 in Section 4.1.1. However, we assume that the model will be extended with additional biochemical species and reactions, as will be the case in Section 7.3.

4.2.4 Drawbacks of the Action Priorities Approach

PAwP has not been designed for multi-scale modelling of biological systems and there are a few disadvantages. These are:

- the introduction of intermediate states, which could be biologically meaningless, i.e. inconsistent with the description of the system modelled. These states are characterised by outgoing transitions labelled by actions with priority higher than 1. In Section 4.2.2 states from 10 to 28 are intermediate states. Some of these states are inconsistent with the definition of the system. For example, state 24 present process $Cell_A$, representing the *food absorbing* phenotype, but only the concentration of **B** is above a threshold, instead of both **A** and **B** as required. Intermediate states can be removed in a second moment merging states that are connected by transitions with action priority higher than 1;

- the use of actions with priorities higher than 1 can produce paths composed of such actions between two biologically meaningful states. If more than one path is possible between two states, these paths could contain different actions. This is a problem if one considers only one of these paths to be correct. For example, consider the following two paths from Section 4.2.2:

$$8 \xrightarrow{e} 25 \xrightarrow{p} 27 \xrightarrow{x} 22 \xrightarrow{q} 24 \xrightarrow{y} 6$$
$$8 \xrightarrow{e} 25 \xrightarrow{q} 16 \xrightarrow{p} 6$$

  The two paths between the two biologically meaningful states 8 and 6 present different sequences of actions. In the first path cellular scale actions x and y are performed, while this is not the case in the second path. In practice we have a non deterministic choice of behaviour at the cellular scale. Assume now that in this case only the second sequence of actions is the correct one. Indeed this is reasonable, because the first sequence contains x, which should be performed only when both **A** and **B** are above their concentration threshold, while this is not the case because action e simply converts one level of concentration of **A** in one of **B**. The operation of specifying which path is the correct one will have to be performed *a posteriori*, once all the paths between two biologically meaningful states have been computed. We note this operation could be automatic. For example one could impose that the shortest path is the correct one.

- PAwP does not present syntactic elements that can be used to define explicitly and without ambiguities distinct scales and actions that work within

and between scales. We have discussed in Section 4.2.2 that the compositionality of a multi-scale model can be improved if actions with priority 1 are used for actions operating within a scale and actions with priority higher than 1 for actions operating between scales. However, nothing in the algebra forces a modeller to apply this good practice. For example, in Section 4.2.3 tissue action $apo1$ synchronises directly with a biochemical process such as $A1_1$.

The disadvantages mentioned above do not prevent a modeller from using PAwP in a multi-scale scenario. The first two can be addressed after all transitions from a state have been computed, while the third is a matter of style of modelling.

4.3 Summary

We began this chapter with a discussion of some of the possible relationships between spatial scales of biological systems. We then addressed the problem of modelling such relationships using two existing process algebras with multi-way synchronisation: simple process algebra and process algebra with priorities. While the limitations of the first became evident quickly, the second proved to be fairly well suited to the task. However, process algebra with priorities has not been designed to model multi-scale scenarios and presents some drawbacks that have been highlighted at the end of this chapter. Although these drawbacks do not in general compromise the use of process algebra with priorities, they nevertheless need to be addressed outside the syntax and the semantics of the algebra.

In the next chapter we introduce a process algebra with hooks that has been designed for multi-scale modelling and addresses the above drawbacks. For completeness, we define a stochastic semantics for process algebra with priorities based on functional rates, in analogy with Section 3.2.1. This can be found in Appendix A.

Chapter 5

Multi-Scale Modelling with Process Algebra with Hooks

In this chapter we introduce process algebra with hooks, a process algebra designed for multi-scale modelling of biological systems. In Section 5.1 we introduce the syntax, the main novel features are *composed actions* and a new *vertical operator*. Together, these features provide explicit modelling of scales and interactions within and between scales. In Section 5.1.1 we give some basic examples using process algebra with hooks and in Sections 5.1.2 and 5.1.3 we give process algebra with hooks versions of the examples introduced in the previous chapter. A comparison between process algebra with hooks and process algebra with priorities is given in Section 5.1.4. Finally, we present a stochastic version of the algebra based on functional rates in Section 5.2, with additional examples in Sections 5.2.4 and 5.2.3.

5.1 Process Algebra with Hooks

A preliminary version of process algebra with hooks (PAH) has been published in (Degasperi and Calder, 2010). In our previous work we followed a *bottom-up* approach where the biochemical scale determines the rates and other scales are abstractions of lower scales. Here we follow a *middle-out* (Noble, 2006) approach where one can begin modelling at any scale, and then relate to higher or lower scales.

The syntax of PAH is:

$$P ::= nil \mid \mathcal{A}[\mathcal{E}].P \mid P + P \mid P \underset{\mathcal{L}}{\bowtie} P \mid P \underset{\mathcal{L}}{\Subset} P \mid A$$

where:

- P is a process, $P \in \mathbb{P}$, with $\mathbb{P}$ the set of processes;
- $\mathcal{L}$, $\mathcal{A}$ and $\mathcal{E}$ are multi-sets of actions, with $\mathcal{L} = (\mathcal{L}', m_{\mathcal{L}})$, $\mathcal{A} = (\mathcal{A}', m_{\mathcal{A}})$ and $\mathcal{E} = (\mathcal{E}', m_{\mathcal{E}})$. Moreover, $\mathcal{L}' \subseteq \mathit{Actions}$, $\mathcal{A}' \subseteq \mathit{Actions} \wedge \mathcal{A} \neq \emptyset$, $\mathcal{E}' \subseteq \mathit{Actions} \wedge |\mathcal{E}'| \leq 1$, with $\mathit{Actions}$ the set of actions;
- $\mathcal{A}[\mathcal{E}]$ is a composed action. Actions in $\mathcal{A}$ are called *layer* actions, while actions in $\mathcal{E}$ are called *hook* actions;
- nil is the deadlock process;
- $\mathcal{A}[\mathcal{E}].P$ expresses the fact that the composed action $\mathcal{A}[\mathcal{E}]$ has to be performed in order to change process $\mathcal{A}[\mathcal{E}].P$ into the new process P;
- $P + P$ expresses the non deterministic choice between two processes. Once one is chosen, the other is discarded;
- $P \underset{\mathcal{L}}{\bowtie} P$ expresses the horizontal cooperation between two independent processes on *the same* scale via the cooperation multi-set $\mathcal{L}$;
- $P \underset{\mathcal{L}}{\Subset} P$ expresses the vertical cooperation between two independent processes on *different* scales via the cooperation multi-set $\mathcal{L}$;
- A is used to recursively define processes, via the agent definition $A \triangleq P$.

Conventions for the notation of actions are as follows. Given a composed action $\mathcal{A}[\mathcal{E}]$, if $|\mathcal{A}| = 1$ or $|\mathcal{E}| = 1$, then set delimiters can be omitted, e.g. if $\mathcal{A} = \{\!|a|\!\}$, then it can be written a. If $\mathcal{E} = \emptyset$ then the hook part of the composed action can be omitted completely, that is $\mathcal{A}[\emptyset]$ can be written $\mathcal{A}$.

The main differences between PAH and SPA of Chapter 3, are the substitution of simple a actions by more complex composed actions $\mathcal{A}[\mathcal{E}]$ and the addition of the vertical cooperation operator $\underset{\mathcal{L}}{\Subset}$. The intended interpretation of these new features is as follows:

- $\mathcal{A}[\mathcal{E}]$ is interpreted as "on this scale perform the actions in multi-set $\mathcal{A}$ all together, while broadcasting actions in multi-set $\mathcal{E}$ to the other scales". It implies that **actions in $\mathcal{A}$ affect the local scale** while **actions in $\mathcal{E}$ affect other scales**. We refer to actions performed by a process as *layer* actions if they belong to multi-set $\mathcal{A}$ and *hook* actions if they belong to multi-set $\mathcal{E}$. Hook actions will be synchronised with layer actions on other scales;

- $\underset{\mathcal{L}}{\bowtie}$ has the role of synchronising layer actions on one side of the operator with hook actions on the other side via actions present in $\mathcal{L}$. It implies that **the process on the left is on a different scale to the process on the right** and that **the actions in $\mathcal{L}$ work between scales**. No hook-with-hook or layer-with-layer action synchronisations are allowed by this operator.

The semantics of PAH is defined by the derivation rules in Figure 5.1.

Rule **Prefix** is an axiom that expresses that process $\mathcal{A}[\mathcal{E}].P$ can become process P via the execution of composed action $\mathcal{A}[\mathcal{E}]$. Although we restrict the set $\mathcal{E}$ in the syntax to be either empty or a singleton, this set can merge with others upon the application of rules **Layer Synchronisation**, **Vertical Synchronisation Left** and **Vertical Synchronisation Right**, producing a multi-set of hooks. We use multi-sets to allow a more general and flexible composition both within and between scales. Example 4 in Section 5.1.1 provides an example of these compositions.

Rules **Choice Left** and **Choice Right** express choice between the execution of composed actions.

Rule **Layer Synchronisation** is a weaker version of the same rule for SPA (Section 3.1). In fact, this rule can be applied even if the labels of the transitions from processes P and Q are not identical: it is only requested that multi-sets $\mathcal{A}$ and $\mathcal{B}$ share at least a name and that this name is also in $\mathcal{L}$. The resulting transition presents the multi-set union of multi-sets of layer actions $\mathcal{A}$ and $\mathcal{B}$, to represent the result of the synchronisation. Conversely, multi-set sum of multi-sets of hooks $\mathcal{E}$ and $\mathcal{F}$ is used to represent the collection of hooks summing the multiplicity of the hook actions. In rules **Asynchronous Left** and **Asynchronous Right**, processes in cooperation can proceed asynchronously only if action set $\mathcal{A}$ does not share actions with cooperation set $\mathcal{L}$.

Prefix

$$\frac{}{\mathcal{A}[\mathcal{E}].P \xrightarrow{\mathcal{A}[\mathcal{E}]} P}$$

Asynchronous Right

$$\frac{Q \xrightarrow{\mathcal{A}[\mathcal{E}]} Q'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{\mathcal{A}[\mathcal{E}]} P \underset{\mathcal{L}}{\bowtie} Q'} \; \mathcal{A} \cap \mathcal{L} = \emptyset$$

Choice Right

$$\frac{Q \xrightarrow{\mathcal{A}[\mathcal{E}]} Q'}{P + Q \xrightarrow{\mathcal{A}[\mathcal{E}]} Q'}$$

Asynchronous Left

$$\frac{P \xrightarrow{\mathcal{A}[\mathcal{E}]} P'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{\mathcal{A}[\mathcal{E}]} P' \underset{\mathcal{L}}{\bowtie} Q} \; \mathcal{A} \cap \mathcal{L} = \emptyset$$

Choice Left

$$\frac{P \xrightarrow{\mathcal{A}[\mathcal{E}]} P'}{P + Q \xrightarrow{\mathcal{A}[\mathcal{E}]} P'}$$

Layer Synchronisation

$$\frac{P \xrightarrow{\mathcal{A}[\mathcal{E}]} P' \quad Q \xrightarrow{\mathcal{B}[\mathcal{F}]} Q'}{P \underset{\mathcal{L}}{\bowtie} Q \xrightarrow{\mathcal{A} \cup \mathcal{B}[\mathcal{E} \uplus \mathcal{F}]} P' \underset{\mathcal{L}}{\bowtie} Q'} \; \mathcal{A} \cap \mathcal{B} \cap \mathcal{L} \neq \emptyset$$

Agent

$$\frac{P \xrightarrow{\mathcal{A}[\mathcal{E}]} P'}{A \xrightarrow{\mathcal{A}[\mathcal{E}]} P'} \; A \triangleq P$$

Vertical Asynchronous Right

$$\frac{Q \xrightarrow{\mathcal{B}[\mathcal{F}]} Q'}{P \underset{\mathcal{L}}{\boxtimes} Q \xrightarrow{\mathcal{B}[\mathcal{F}]} P \underset{\mathcal{L}}{\boxtimes} Q'} \; \begin{array}{l} \mathcal{B} \cap \mathcal{L} = \emptyset \wedge \\ \neg(\exists P \xrightarrow{\mathcal{A}[\mathcal{E}]} P'. \\ \mathcal{A} \subseteq \mathcal{F} \cap \mathcal{L}) \end{array}$$

Vertical Asynchronous Left

$$\frac{P \xrightarrow{\mathcal{A}[\mathcal{E}]} P'}{P \underset{\mathcal{L}}{\boxtimes} Q \xrightarrow{\mathcal{A}[\mathcal{E}]} P' \underset{\mathcal{L}}{\boxtimes} Q} \; \begin{array}{l} \mathcal{A} \cap \mathcal{L} = \emptyset \wedge \\ \neg(\exists Q \xrightarrow{\mathcal{B}[\mathcal{F}]} Q'. \\ \mathcal{B} \subseteq \mathcal{E} \cap \mathcal{L}) \end{array}$$

Vertical Synchronisation Left

$$\frac{P \xrightarrow{\mathcal{A}[\mathcal{E}]} P' \quad Q \xrightarrow{\mathcal{B}[\mathcal{F}]} Q'}{P \underset{\mathcal{L}}{\boxtimes} Q \xrightarrow{\mathcal{A} \cup \mathcal{B}[(\mathcal{E} \setminus \mathcal{B}) \uplus \mathcal{F}]} P' \underset{\mathcal{L}}{\boxtimes} Q'} \; \begin{array}{l} \mathcal{B} \subseteq \mathcal{E} \cap \mathcal{L} \; \wedge \; \neg(\exists Q \xrightarrow{\mathcal{B}'[\mathcal{F}']} Q''. \\ (\mathcal{B}' \subseteq \mathcal{E} \cap \mathcal{L}) \wedge (|\mathcal{B}'| > |\mathcal{B}|)) \end{array}$$

Vertical Synchronisation Right

$$\frac{P \xrightarrow{\mathcal{A}[\mathcal{E}]} P' \quad Q \xrightarrow{\mathcal{B}[\mathcal{F}]} Q'}{P \underset{\mathcal{L}}{\boxtimes} Q \xrightarrow{\mathcal{A} \cup \mathcal{B}[(\mathcal{F} \setminus \mathcal{A}) \uplus \mathcal{E}]} P' \underset{\mathcal{L}}{\boxtimes} Q'} \; \begin{array}{l} \mathcal{A} \subseteq \mathcal{F} \cap \mathcal{L} \; \wedge \; \neg(\exists P \xrightarrow{\mathcal{A}'[\mathcal{E}']} P''. \\ (\mathcal{A}' \subseteq \mathcal{F} \cap \mathcal{L}) \wedge (|\mathcal{A}'| > |\mathcal{A}|)) \end{array}$$

Figure 5.1: Semantics of process algebra with hooks. Union of multi-sets is denoted by $\cup$ and sum of multi-sets is denoted by $\uplus$.

The behaviour induced by the $\overset{\mathcal{E}}{\underset{\mathcal{L}}{\bowtie}}$ operator is regulated by the rules **Vertical Synchronisation Left**, **Vertical Synchronisation Right**, **Vertical Asynchronous Left** and **Vertical Asynchronous Right**. Rules differing in the name only by **Left** and **Right** are symmetric, so we explain only one of them. In **Vertical Synchronisation Left**, the synchronisation is between the multi-set of hook actions on the left hand side ($\mathcal{E}$) and the multi-set of layer actions on the right hand side ($\mathcal{B}$), via actions in the cooperation multi-set $\mathcal{L}$. More specifically, some inter-scale actions in $\mathcal{E}$ are interpreted by another scale via $\mathcal{B}$. For this to happen we impose in the side rule that $\mathcal{B}$ must be included in both $\mathcal{E}$ and $\mathcal{L}$. The resulting transition presents the multi-set union of multi-sets $\mathcal{A}$ and $\mathcal{B}$, while $\mathcal{B}$ is subtracted from $\mathcal{E}$ to represent the fact that some of the hook actions of the left hand side have been used. Multi-set sum is used between $\mathcal{E} \setminus \mathcal{B}$ and $\mathcal{F}$ to collect the remaining hook actions. It may be that more than one transition from Q presents a multi-set of suitable layer actions $\mathcal{B}$. In this case, we consider the largest $\mathcal{B}$ multi-sets, imposed by the side condition, which states that there is no other transition from Q which presents a multi-set of layer actions $\mathcal{B}'$ included in both $\mathcal{E}$ and $\mathcal{L}$ that is larger than $\mathcal{B}$. This gives the possibility to the modeller to choose how the model should behave when multiple hook actions are offered in a single transition (see Examples 3 and 4 in Section 5.1.1). Consider now the inference rule **Vertical Asynchronous Left**. In this case, we allow a single process to transition asynchronously only if there are no actions in the multi-set $\mathcal{A}$ which are also contained in $\mathcal{L}$. This is because the actions in the vertical cooperation set $\mathcal{L}$ are hooks and if $\mathcal{A} \cap \mathcal{L} \neq \emptyset$ then $\mathcal{A}$ contains hooks and the transition is not intended to be used asynchronously, but only to respond to hook actions that might arise from transitions from Q. Moreover, this rule can only be applied if no transitions from Q present a multi-set of layer actions $\mathcal{B}$ suitable for synchronisation, i.e. included in both $\mathcal{E}$ and $\mathcal{L}$.

In the next sections we illustrate the use of PAH with a series of basic examples, followed by examples taken from the previous chapter.

5.1.1 Process Algebra with Hooks: Basic Examples

Example 1. The behaviour of a cell depends on the concentration of species **M**. Let M_i ($i = 0, 1, 2, 3$) be the processes representing species **M** with concentration level i. Let $Cell_0$ and $Cell_1$ be processes representing two distinct behaviours (the phenotypes) of the cell.

$$M_0 \triangleq a.M_1 \qquad Cell_0 \triangleq x.Cell_1$$
$$M_1 \triangleq a[x].M_2 + b.M_0 \qquad Cell_1 \triangleq y.Cell_0$$
$$M_2 \triangleq a.M_3 + b[y].M_1$$
$$M_3 \triangleq b.M_2$$

The initial state is:

$$M_1 \underset{\{|x,y|\}}{\bowtie} Cell_0$$

The above processes can be represented with the following graphical representation:

$$Cell_0 \underset{y}{\overset{x}{\rightleftarrows}} Cell_1$$

$$M_0 \underset{b}{\overset{a}{\rightleftarrows}} M_1 \underset{b[y]}{\overset{a[x]}{\rightleftarrows}} M_2 \underset{b}{\overset{a}{\rightleftarrows}} M_3$$

Actions a and b represent biochemical reactions that increase or decrease, respectively, the concentration of **M**. The behaviour of the cell changes when the concentration of **M** passes a threshold. In this case, the action x denotes $Cell_0$ becomes $Cell_1$, when M_1 becomes M_2; and conversely $Cell_1$ becomes $Cell_0$ with action y when M_2 becomes M_1. The composed action $a[x]$ carries two distinct pieces of information: a means the biochemical reaction R_a has happened, while x means a change at the cellular scale has been triggered. We do not represent the execution of $a[x]$ as an interleaving of the action names a and x. Instead, $a[x]$ generates a single labelled transition, thus:

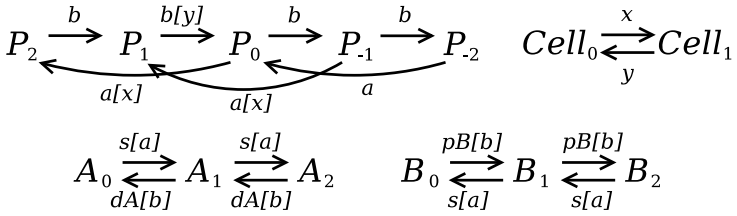
$$M_1 \underset{\{|x,y|\}}{\bowtie} Cell_0 \xrightarrow{\{|a,x|\}[\emptyset]} M_2 \underset{\{|x,y|\}}{\bowtie} Cell_1$$

Without the cell process $Cell_0$, the same transition is:

$$M_1 \xrightarrow{a[x]} M_2$$

In this transition, the hook x is exposed, but not observed by any process.

Example 2. More complex relations between scales can be described. In this example, a change of behaviour of a cell is triggered when the concentration of molecule **A** exceeds the concentration of molecule **B**. Processes P_i can be used to count the difference between the concentration levels of **A** and **B**. The graphical representation of the processes is given by:



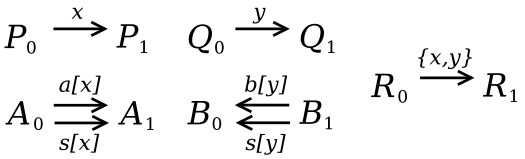
The initial state is:

$$((A_1 \underset{\{\!|s|\!\}}{\bowtie} B_2) \underset{\{\!|a,b|\!\}}{\bowtie} P_{-1}) \underset{\{\!|x,y|\!\}}{\bowtie} Cell_0$$

Species **A** can degrade (dA), **B** can be produced (pB), while both **A** and **B** can synchronise (on s) so that a level of **B** is converted into a level of **A**. Processes P_i, $i \in \{-2, \ldots, 2\}$, represent the difference between the current level of **A** and of **B**, while a and b actions represent events that make this difference increase by 2 and decrease by 1 respectively. An example transition is:

$$((A_1 \underset{\{\!|s|\!\}}{\bowtie} B_2) \underset{\{\!|a,b|\!\}}{\bowtie} P_{-1}) \underset{\{\!|x,y|\!\}}{\bowtie} Cell_0 \xrightarrow{\{\!|s,a,x|\!\}[\emptyset]} ((A_2 \underset{\{\!|s|\!\}}{\bowtie} B_1) \underset{\{\!|a,b|\!\}}{\bowtie} P_1) \underset{\{\!|x,y|\!\}}{\bowtie} Cell_1$$

Example 3. If a scale triggers more than one hook action, these hook actions can be observed individually by multiple observers or together by a single observer. Consider the following processes:



Processes A_0 and B_1 can produce the following transition:

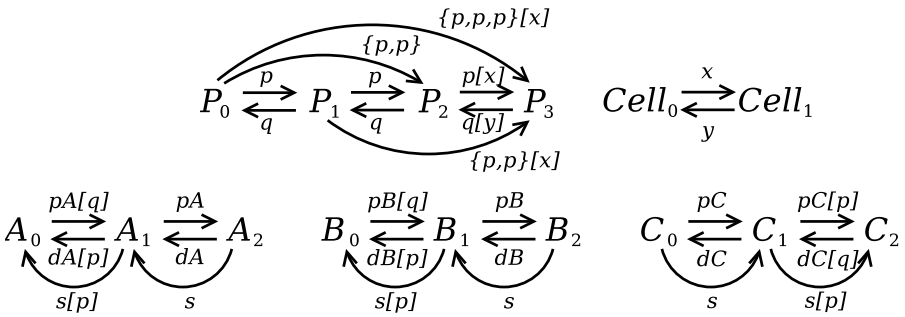
$$A_0 \underset{\{\!|s|\!\}}{\bowtie} B_1 \xrightarrow{s[\{\!|x,y|\!\}]} A_1 \underset{\{\!|s|\!\}}{\bowtie} B_0$$

In this case, two different instances of s synchronise, their set of hook actions merge in the resulting activity. Now consider the addition of processes P_0, Q_0 and R_0. Two possible examples of transition are:

$$(A_0 \underset{\{\!|x|\!\}}{\bowtie} P_0) \underset{\{\!|s|\!\}}{\bowtie} (B_1 \underset{\{\!|y|\!\}}{\bowtie} Q_0) \xrightarrow{\{\!|s,x,y|\!\}[\emptyset]} (A_1 \underset{\{\!|x|\!\}}{\bowtie} P_1) \underset{\{\!|s|\!\}}{\bowtie} (B_0 \underset{\{\!|y|\!\}}{\bowtie} Q_1)$$
$$(A_0 \underset{\{\!|s|\!\}}{\bowtie} B_1) \underset{\{\!|x,y|\!\}}{\bowtie} R_0 \xrightarrow{\{\!|s,x,y|\!\}[\emptyset]} (A_1 \underset{\{\!|s|\!\}}{\bowtie} B_0) \underset{\{\!|x,y|\!\}}{\bowtie} R_1$$

In the first transition, hook actions x and y are observed individually by processes P_0 and Q_0. If only hook action x were present, it would still be observed by P_0 and the same for y with Q_0. In the second transition, hook actions x and y are observed at the same time by process R_0. This can happen only if both x and y are present.

Example 4. In some cases, a scale may trigger a multi-set of hook actions. Consider the following example. A cell contains three biochemical species **A**, **B** and **C**. The cell changes its behaviour if the biochemical scale reaches a specific configuration, that is when the concentrations of **A** and **B** are low and when the concentration of **C** is high. Species **A**, **B** and **C** are produced (actions pA, pB and pC) and degrade (actions dA, dB and dC) in the cell. Finally, species **A**, **B** and **C** are involved in the biochemical reaction $R_s : \mathbf{A} + \mathbf{B} \rightarrow \mathbf{C}$. The model is defined as:



The initial state is:

$$((A_1 \underset{\{|s|\}}{\bowtie} B_1 \underset{\{|s|\}}{\bowtie} C_1) \underset{\{|p,p,p,q|\}}{\bowtie} P_0) \underset{\{|x,y|\}}{\bowtie} Cell_0$$

Again we use processes P_i, $i \in \{0, \ldots, 3\}$, to count the concentration required for change at the cellular scale. In this particular case, a single transition can involve more than one identical hook action (i.e. p). The transition is the following:

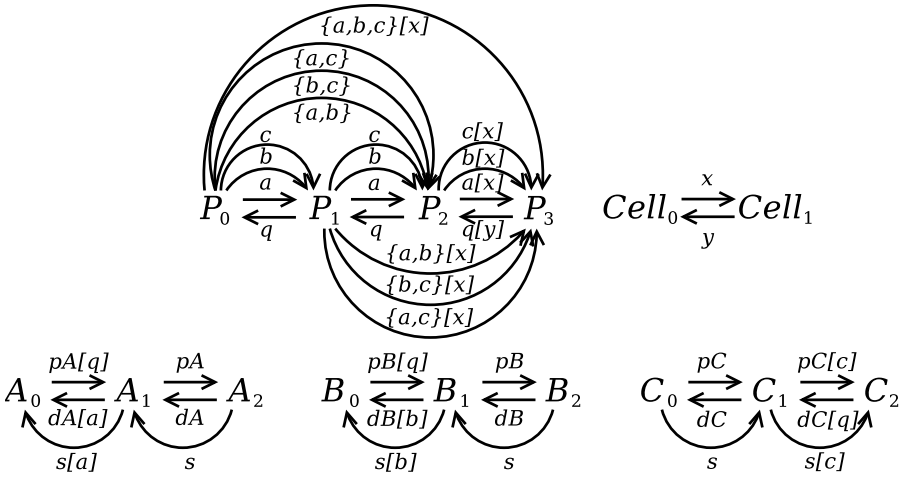
$$\begin{array}{c}((A_1 \underset{\{|s|\}}{\bowtie} B_1 \underset{\{|s|\}}{\bowtie} C_1) \underset{\{|p,p,p,q|\}}{\bowtie} P_0) \underset{\{|x,y|\}}{\bowtie} Cell_0 \xrightarrow{\{|s,p,p,p,x|\}[\emptyset]} \\ ((A_0 \underset{\{|s|\}}{\bowtie} B_0 \underset{\{|s|\}}{\bowtie} C_2) \underset{\{|p,p,p,q|\}}{\bowtie} P_3) \underset{\{|x,y|\}}{\bowtie} Cell_1\end{array}$$

Another transition of the derivation graph generated by this model is:

$$\begin{array}{c}((A_2 \underset{\{|s|\}}{\bowtie} B_1 \underset{\{|s|\}}{\bowtie} C_1) \underset{\{|p,p,p,q|\}}{\bowtie} P_1) \underset{\{|x,y|\}}{\bowtie} Cell_0 \xrightarrow{\{|s,p,p,x|\}[\emptyset]} \\ ((A_1 \underset{\{|s|\}}{\bowtie} B_0 \underset{\{|s|\}}{\bowtie} C_2) \underset{\{|p,p,p,q|\}}{\bowtie} P_3) \underset{\{|x,y|\}}{\bowtie} Cell_1\end{array}$$

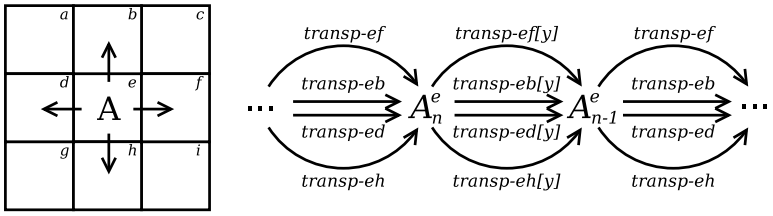
The use of multi-sets allows for a compact definition of this model, where we use the same action p to indicate that either **A**, **B** or **C** have reached the concentration required for a change in behaviour of the cell.

Without multi-sets, an alternative definition of the same model is as follows:



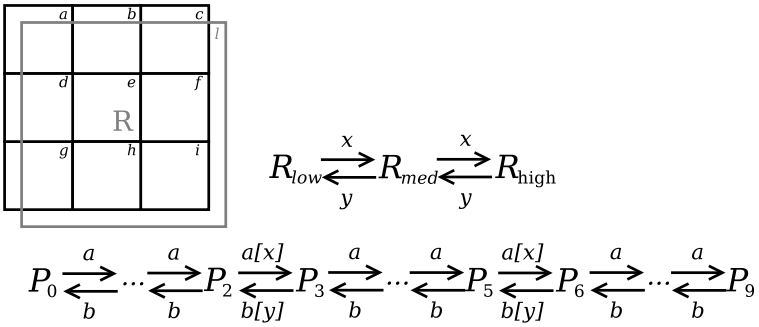
The above version of the model uses actions a, b and c in place of p creating a combinatorial problem. As a result, the derivation graph of process P_0 presents 19 transitions instead of eight.

Example 5. The positioning of hook actions on actions at the biochemical scale is particularly useful when geometrical space is considered. Let A^e_n denote the process representing a concentration level n of species **A** in region R_e. Concentration can migrate to and from region R_e and many different transport actions will have the same effect of lowering or increasing the concentration of **A** in one region, as shown in the following diagram (only outgoing transport shown):



The concentration of **A** is decreased, from A^e_n to A^e_{n-1}, through a transport action of the form *transp-es*, $s \in \{b, d, f, h\}$. Correspondingly, at region R_s, the concentration of **A** increases, from A^s_m to A^s_{m+1}. If we want to denote that a threshold is crossed when passing from level n to $n-1$ of **A** at R_e, we can add a hook action to the four transport actions, *transp-es*, obtaining *transp-es*$[y]$.

Example 6. In this example we show how to abstract multiple regions to a single region, with respect to a specific property. Consider an area R_l of 3×3 regions, labelled from R_a to R_i. We ignore detail of the biochemical reactions, but assume that at some point each of these locations can become *infected*, exposing hook action a, or they can *recover*, exposing hook action b. We are not interested in which region has changed its status, only the number infected in R_l. A scale that represents the degree of infection of area R_l is defined by three processes, R_{low}, R_{med} and R_{high}.



Processes P_i, $i \in \{0, \ldots, 9\}$ are used to count the number of infected regions. If the number of infected regions is between 0 and 2, the degree of infection is low; between 3 and 5 it is medium; larger than 5 it is high. Hooks x and y identify transitions between stages of infection.

5.1.2 Process Algebra with Hooks and a Three Layers Example

In Section 4.2.2 we modelled three scales using PAwP. We defined a biochemical scale, a cellular scale and an intermediate layer which we used to count how many of biochemical species **A** and **B** presented their concentration above a threshold. This produced a considerable number of intermediate states, i.e. states from which only actions with priority higher than 1 are possible. Here, we show how we can rewrite the same example in PAH, and generate only states that are biologically meaningful. In particular, each state corresponds to a different combination of concentration levels of the two biochemical species **A** and **B**. The processes are:

$$
\begin{array}{lll}
A_0 \triangleq a.A_1 + f.A_1 & B_0 \triangleq c.B_1 + e.B_1 & P_0 \triangleq p.P_1 \\
A_1 \triangleq a[p].A_2 + b.A_0 & B_1 \triangleq c[p].B_2 + d.B_0 & P_1 \triangleq q.P_0 + p[x].P_2 + \{\!|p,q|\!\}.P_1 \\
\quad +f[p].A_2 + e.A_0 & \quad +e[p].B_2 + f.B_0 & P_2 \triangleq q[y].P_1 \\
A_2 \triangleq b[q].A_1 + e[q].A_1 & B_2 \triangleq d[q].B_1 + f[q].B_1 & \\
Cell_M \triangleq move.Cell_M + x.Cell_A & Cell_A \triangleq absorb.Cell_A + y.Cell_M &
\end{array}
$$

The most interesting difference between this and the PAwP description in Section 4.2.2 is the additional choice of $\{\!|p,q|\!\}.P_1$ of process P_1. This additional choice allows the modeller to determine what process P_1 should do if two hook actions originate from the biochemical scale. The initial state of the model is given by:

$$((A_0 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M$$

where $\mathcal{L} = \{\!|e,f|\!\}$, $\mathcal{H} = \{\!|p,q|\!\}$ and $\mathcal{K} = \{\!|x,y|\!\}$. The states of the model reachable from the initial state are:

$$
\begin{array}{ll}
1\text{: } ((A_0 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M & 2\text{: } ((A_0 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M \\
3\text{: } ((A_1 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M & 4\text{: } ((A_1 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M \\
5\text{: } ((A_0 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M & 6\text{: } ((A_1 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M \\
7\text{: } ((A_2 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M & 8\text{: } ((A_2 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\bowtie} P_1) \underset{\mathcal{K}}{\bowtie} Cell_M \\
9\text{: } ((A_2 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\bowtie} P_2) \underset{\mathcal{K}}{\bowtie} Cell_A &
\end{array}
$$

The nine states correspond exactly to the first nine states of the PAwP model, Section 4.2.2. States and transitions of the model are shown in Figure 5.2. Interesting transitions are $8 \xrightarrow{\{\!|c,p,x|\!\}} 9$, which involves all three layers at once, and $8 \xrightarrow{\{\!|e,p,q|\!\}} 6$, where the concentration of **A** and **B** cross their respective threshold at the same time, from different directions, with no effect to the cellular scale.

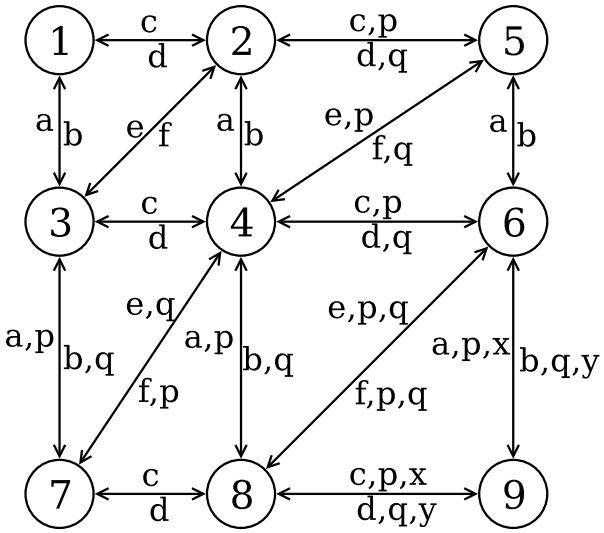


Figure 5.2: Reachable states and transitions generated by the example in Section 5.1.2.

5.1.3 Process Algebra with Hooks and Tissue Growth with Biochemistry

In this section we show how the example introduced in Section 4.2.3 can be modelled using PAH. Because PAH adopts an explicit distinction between actions operating within a scale and between scales, new actions have to be introduced. In particular, a direct synchronisation between biochemical and tissue scales via actions such as $apo1$ or $mito21$ would be prevented by the vertical operator $\overset{\boxtimes}{\underset{\mathcal{L}}{}}$. Instead, we introduce inter-scale actions $bioff1$ and $bioon1$ to communicate the effects of $apo1$ and $mito21$ across scales. Processes can be rewritten as follows:

$$NA0 \triangleq bioon0.A0_0$$
$$A0_0 \triangleq bioff0.NA0 + a0.A0_1$$
$$A0_1 \triangleq bioff0.NA0 + a0[mitoon0].A0_2 + b0.A0_0$$
$$A0_2 \triangleq bioff0.NA0 + b0[mitoff0].A0_1$$

$$Empty0 \triangleq mito10[bioon0].Tissueoff0$$
$$Tissueoff0 \triangleq apo0[bioff0].Empty0 + mitoon0.Tissueon0$$
$$Tissueon0 \triangleq apo0[bioff0].Empty0 + [mito01].Tissueon0$$
$$+ mitoff0.Tissueoff0$$

$$NA1 \triangleq bioon1.A1_0$$
$$A1_0 \triangleq biooff1.NA1 + a1.A1_1$$
$$A1_1 \triangleq biooff1.NA1 + a1[mitoon1].A1_2 + b1.A1_0$$
$$A1_2 \triangleq biooff1.NA1 + b1[mitooff1].A1_1$$

$$Empty1 \triangleq mito21[bioon1].Tissueoff1 + mito01[bioon1].Tissueoff1$$
$$Tissueoff1 \triangleq apo1[biooff1].Empty1 + mitoon1.Tissueon1$$
$$Tissueon1 \triangleq apo1[biooff1].Empty1 + mito12.Tissueon1 + mito10.Tissueon1 + mitooff1.Tissueoff1$$

$$NA2 \triangleq bioon2.A2_0$$
$$A2_0 \triangleq biooff2.NA2 + a2.A2_1$$
$$A2_1 \triangleq biooff2.NA2 + a2[mitoon2].A2_2 + b2.A2_0$$
$$A2_2 \triangleq biooff2.NA2 + b2[mitooff2].A2_1$$

$$Empty2 \triangleq mito12[bioon2].Tissueoff2$$
$$Tissueoff2 \triangleq apo2[biooff2].Empty2 + mitoon2.Tissueon2$$
$$Tissueon2 \triangleq apo2[biooff2].Empty2 + mito21.Tissueon2 + mitooff2.Tissueoff2$$

The initial state of the model is defined by the following process:

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$

with $\mathcal{L} = \{\!|bioon0,\ biooff0,\ bioon1,\ biooff1,\ bioon2,\ biooff2,\ mitoon0,\ mitooff0,\ mitoon1,\ mitooff1,\ mitoon2,\ mitooff2|\!\}$, $\mathcal{K} = \{\!|mito10,\ mito01|\!\}$, $\mathcal{K} = \{\!|mito21,\ mito12|\!\}$.

A graphical representation of processes $NA1$ and $Empty1$ is given in Figure 5.3.

Examples of valid transitions are:

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$
$$\xrightarrow{\{\!|b2, mitooff2|\!\}} (NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_1) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueoff2)$$

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$
$$\xrightarrow{\{\!|apo1, biooff1|\!\}} (NA0 \underset{\emptyset}{\bowtie} NA1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Empty1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$

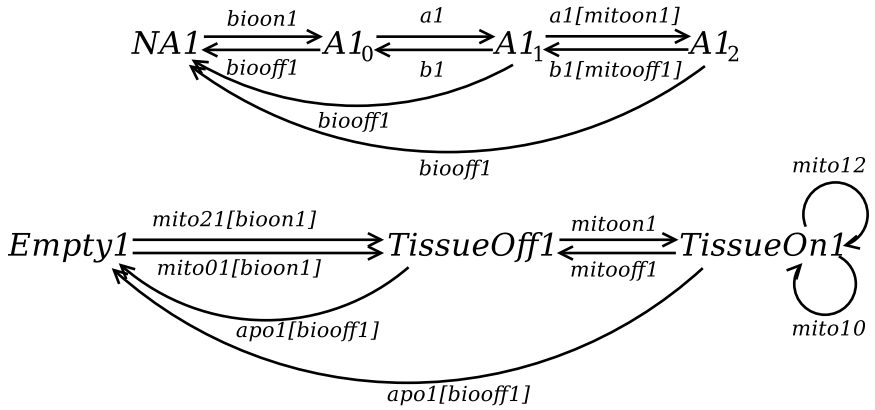


Figure 5.3: Graphical representation of processes $NA1$ and $Empty1$.

An example of an action that is not allowed from the initial state is:

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$
$$\xrightarrow{\{\!|mitooff2|\!\}} (NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueoff2)$$

The reason is that rule **Vertical Synchronisation Right** does not allow $mitooff2$ to be executed, because $mitooff2 \in \mathcal{L}$. The only way $mitooff2$ can be executed is via synchronisation with a hook action performed on the left hand side of $\underset{\mathcal{L}}{\bowtie}$.

5.1.4 Comparison of Process Algebra with Hooks and Process Algebra with Priorities

So far we have seen how PAH can be employed to model biological scales and interactions between scales. The most important difference between PAH and PAwP (Section 4.2) is their different approach to the use of actions to represent events within and between scales in a multi-scale setting. We sum up this difference in Figure 5.4. On the left of the figure, priorities are attached to actions. Actions with priority 1 are considered local while actions with priority 2 are broadcasted to the other scales. On the right of the figure, actions are composed, where the first component, a, interacts locally, while the second component, x, interacts with other scales.

We contend that PAH overcomes the drawbacks of PAwP (as discussed in Section 4.2.4) as follows:

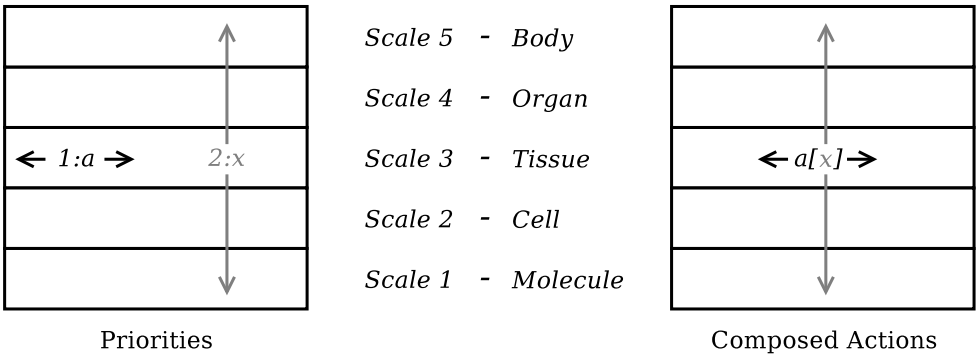


Figure 5.4: Two approaches to the use of actions in process algebra in a multi-scale setting.

- no intermediate states are introduced. In PAwP, actions with priority higher than 1 are used to communicate the effect of an event happening at a scale to other scales. As the event and its inter-scale effect are modelled by two distinct actions, this creates intermediate states. In PAH, an event and its inter-scale effect are fused into a single composed action which generates a single transition;

- more control is given to the modeller concerning how to model the response of a scale to events generating multiple inter-scale effects at the same time. In PAwP, multiple inter-scale effects are modelled by the interleaving of actions with priority higher than 1. As we discussed in Section 4.2.4, this is a problem if the modeller needs to specify which interleaving is the correct one. In fact this cannot be specified within the algebra and requires a dedicated implementation. In PAH, multiple inter-scale events are modelled as a multi-set of hook actions. With this approach, the modeller can specify whether multiple inter-scale effects at a single time produce a different response with respect to the single effects taken individually. This is possible within the algebra by using sets of actions to synchronise with multi-sets of hooks;

- scales and interactions within and between scales can be defined without ambiguities. While PAwP does not present syntactic elements that abstract explicitly the concepts of scales and interactions within and between scales, PAH provides them. The syntactic elements are composed actions and a new vertical synchronisation operator.

In the next section we introduce a stochastic semantics for PAH.

5.2 Stochastic Semantics for Process Algebra with Hooks

In this section we define a stochastic semantics for PAH, in analogy with Section 3.2.1. Before we give a formal definition, we illustrate the main challenges with an example.

Consider Example 2 of Section 4.1.1. In PAH agent definitions are as follows:

$$A_0 \triangleq a.A_1 + f.A_1 \quad A_1 \triangleq a.A_2 + f.A_2 + b.A_0 + e.A_0 \quad A_2 \triangleq b.A_1 + e.A_1$$

$$B_0 \triangleq c.B_1 + e.B_1 \quad B_1 \triangleq c[x].B_2 + e[x].B_2 + d.B_0 + f.B_0 \quad B_2 \triangleq d[y].B_1 + f[y].B_1$$

$$Cell_M \triangleq move.Cell_M + x.Cell_A$$
$$Cell_A \triangleq absorb.Cell_A + y.Cell_M$$

The initial state is:

$$(A_1 \underset{\{|e,f|\}}{\bowtie} B_1) \underset{\{|x,y|\}}{\bowtie} Cell_M$$

An appropriate stochastic semantics for PAH should construct an environment Γ to evaluate the appropriate functional rates. This should be done in analogy with our approach for sSPA in Section 3.2.1.

We assign variables and values to agents:

$$Var(A_0) = A \quad Var(B_0) = B \quad Var(Cell_M) = cell$$
$$Val(A_0) = 0 \quad Val(B_0) = 0 \quad Val(Cell_M) = 1$$
$$Var(A_1) = A \quad Var(B_1) = B \quad Var(Cell_A) = cell$$
$$Val(A_1) = 1 \quad Val(B_1) = 1 \quad Val(Cell_A) = 2$$
$$Var(A_2) = A \quad Var(B_2) = B$$
$$Val(A_2) = 2 \quad Val(B_2) = 2$$

Functional rates and sets of participants for actions are:

$$
\begin{array}{llll}
f_a = k_a/h & p_a = \{A\} & f_d = (k_d * B * h)/h & p_d = \{B\} \\
f_b = (k_b * A * h)/h & p_b = \{A\} & f_e = (k_e * A * h)/h & p_e = \{A, B\} \\
f_c = k_c/h & p_c = \{B\} & f_f = (k_f * B * h)/h & p_f = \{A, B\}
\end{array}
$$

Two valid derivations for the above example should be:

$$
\begin{aligned}
&(A_1 \underset{\emptyset}{\bowtie} B_2) \underset{\{\!|x,y|\!\}}{\boxtimes} Cell_M \xrightarrow{(b,\Gamma)} (A_0 \underset{\emptyset}{\bowtie} B_2) \underset{\{\!|x,y|\!\}}{\boxtimes} Cell_M \\
&(A_1 \underset{\emptyset}{\bowtie} B_2) \underset{\{\!|x,y|\!\}}{\boxtimes} Cell_M \xrightarrow{(\{\!|a,x|\!\},\Gamma')} (A_2 \underset{\emptyset}{\bowtie} B_2) \underset{\{\!|x,y|\!\}}{\boxtimes} Cell_A
\end{aligned}
$$

where $\Gamma = \{(A, 1)\}$ and $\Gamma' = \{(A, 1)\}$. Notice that in the second derivation Γ' should be used to evaluate f_a, while the set of actions performed is $\{\!|a, x|\!\}$. This implies that we need a new mechanism to identify the correct functional rate to use, in this case f_a and not f_x or $f_{\{\!|a,x|\!\}}$. Using the additional environment $\Gamma = \{(h, 1), (k_a, 1), (k_b, 1),\ (k_c, 1),\ (k_d, 1),\ (k_e, 1),\ (k_f, 1)\ \}$, the two examples of rated transitions should be:

$$
\begin{aligned}
&(A_1 \underset{\emptyset}{\bowtie} B_2) \underset{\{\!|x,y|\!\}}{\boxtimes} Cell_M \xrightarrow{(b,1)} (A_0 \underset{\emptyset}{\bowtie} B_2) \underset{\{\!|x,y|\!\}}{\boxtimes} Cell_M \\
&(A_1 \underset{\emptyset}{\bowtie} B_2) \underset{\{\!|x,y|\!\}}{\boxtimes} Cell_M \xrightarrow{(\{\!|a,x|\!\},1)} (A_2 \underset{\emptyset}{\bowtie} B_2) \underset{\{\!|x,y|\!\}}{\boxtimes} Cell_A
\end{aligned}
$$

We suggest that it should be responsibility of the modeller to ensure that every action set $\mathcal{A}$ on a valid transition $M \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M'$ contains one and only one action a such that $f_a \in \mathbb{F}$. If this is not the case for a transition, then rating of such transition should not be possible.

5.2.1 Stochastic Process Algebra with Hooks

In analogy with Sections 3.2.1 we proceed to the definition of a new syntax and semantics for PAH. The syntax is given by:

$$
\begin{aligned}
D &::= nil \mid \mathcal{A}[\mathcal{E}].A \mid D + D \\
M &::= A \mid M \underset{\mathcal{L}}{\bowtie} M \mid M \underset{\mathcal{L}}{\boxtimes} M
\end{aligned}
$$

where:

- D is a *definition* process, $D \in \mathbb{P}_d$, while M is a *model* process, $M \in \mathbb{P}_m$. Definition and model processes are disjoint and are both processes, i.e. $\mathbb{P}_d \cup \mathbb{P}_m = \mathbb{P}$ and $\mathbb{P}_d \cap \mathbb{P}_m = \emptyset$;

- Agents are defined as $A \triangleq D$, that is we use definition processes to define the behaviour of agents;
- a model is defined by a model process M, which in turn is either an agent A or a cooperation between model processes $M \underset{\mathcal{L}}{\bowtie} M$;
- action execution $\mathcal{A}[\mathcal{E}].A$ is always followed by an agent A. This ensures that at any time the state of a model will be constituted of cooperations of agents;
- functions $Var(A)$ and $Val(A)$ must be defined for each agent A, with $Var(A) \in Names$, $Val(A) \in \mathbb{R}$ and $Names$ the set of parameter names.

The stochastic semantics of the new syntax is presented in Figure 5.5. We refer to this new process algebra as *stochastic process algebra with hooks* (sPAH).

In analogy with sSPA, we use the semantics to construct environments that will be used by the rating routines to derive a rated derivation graph. In Section 5.2.2 we define constraints on a sPAH model that ensure that the computation of rates is always correct, in analogy with Section 3.2.3 for sSPA. These constraints are effective also because in the derivation rule **Vertical Synchronisation Left** we discard Γ_2 (in **Vertical Synchronisation Right** we discard Γ_1), a mechanism that guarantees that we can use a single set of participants p_a, as for sSPA.

We introduce now definitions necessary to define the derivation graph for sPAH processes, in analogy with Sections 3.2.1. In addition, we give a semantics of sPAH in terms of multi-set of *moves*. This semantics is necessary in Chapter 6, where the multiplicity of the transitions between sPAH processes is used in the definition of equivalence relations.

Definition 5.1 *Activity.* The pair $(\mathcal{A}[\mathcal{E}], \Gamma)$ such that $\mathcal{A}, \mathcal{E} \subseteq Actions$ and $\Gamma \subseteq Names \times \mathbb{R}$ is called an activity.

Definition 5.2 *One step derivative.* If $M \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M'$ then M is a one step derivative of M.

Definition 5.3 *Derivative.* If $M_i \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} \ldots \xrightarrow{(\mathcal{A}'[\mathcal{E}'],\Gamma')} M_j$ then M_j is a derivative of M_i.

Prefix

$$\frac{}{\mathcal{A}[\mathcal{E}].A \xrightarrow{\mathcal{A}[\mathcal{E}]} A}$$

Agent

$$\frac{D \xrightarrow{\mathcal{A}[\mathcal{E}]} A' \quad A \triangleq D}{A \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} A' \quad \wedge\, \Gamma = \{(Var(A), Val(A))\}}$$

Choice Left

$$\frac{D_1 \xrightarrow{\mathcal{A}[\mathcal{E}]} A}{D_1 + D_2 \xrightarrow{\mathcal{A}[\mathcal{E}]} A}$$

Asynchronous Left

$$\frac{M_1 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M_1'}{M_1 \underset{\mathcal{L}}{\bowtie} M_2 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M_1' \underset{\mathcal{L}}{\bowtie} M_2} \; \mathcal{A} \cap \mathcal{L} = \emptyset$$

Choice Right

$$\frac{D_2 \xrightarrow{\mathcal{A}[\mathcal{E}]} A}{D_1 + D_2 \xrightarrow{\mathcal{A}[\mathcal{E}]} A}$$

Asynchronous Right

$$\frac{M_2 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M_2'}{M_1 \underset{\mathcal{L}}{\bowtie} M_2 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M_1 \underset{\mathcal{L}}{\bowtie} M_2'} \; \mathcal{A} \cap \mathcal{L} = \emptyset$$

Layer Synchronisation

$$\frac{M_1 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma_1)} M_1' \quad M_2 \xrightarrow{(\mathcal{B}[\mathcal{F}],\Gamma_2)} M_2'}{M_1 \underset{\mathcal{L}}{\bowtie} M_2 \xrightarrow{(\mathcal{A}\cup\mathcal{B}[\mathcal{E}\uplus\mathcal{F}],\Gamma_1\cup\Gamma_2)} M_1' \underset{\mathcal{L}}{\bowtie} M_2'} \; \mathcal{A} \cap \mathcal{B} \cap \mathcal{L} \neq \emptyset$$

Vertical Asynchronous Left

$$\frac{M_1 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M_1'}{M_1 \underset{\mathcal{L}}{\boxtimes} M_2 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M_1' \underset{\mathcal{L}}{\boxtimes} M_2} \; \begin{array}{l} \mathcal{A} \cap \mathcal{L} = \emptyset \wedge \\ \neg(\exists M_2 \xrightarrow{(\mathcal{B}[\mathcal{F}],\Gamma')} M_2'. \\ \mathcal{B} \subseteq \mathcal{E} \cap \mathcal{L}) \end{array}$$

Vertical Asynchronous Right

$$\frac{M_2 \xrightarrow{(\mathcal{B}[\mathcal{F}],\Gamma)} M_2'}{M_1 \underset{\mathcal{L}}{\boxtimes} M_2 \xrightarrow{(\mathcal{B}[\mathcal{F}],\Gamma)} M_1 \underset{\mathcal{L}}{\boxtimes} M_2'} \; \begin{array}{l} \mathcal{B} \cap \mathcal{L} = \emptyset \wedge \\ \neg(\exists M_1 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma')} M_1'. \\ \mathcal{A} \subseteq \mathcal{F} \cap \mathcal{L}) \end{array}$$

Vertical Synchronisation Left

$$\frac{M_1 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma_1)} M_1' \quad M_2 \xrightarrow{(\mathcal{B}[\mathcal{F}],\Gamma_2)} M_2'}{M_1 \underset{\mathcal{L}}{\boxtimes} M_2 \xrightarrow{(\mathcal{A}\cup\mathcal{B}[(\mathcal{E}\setminus\mathcal{B})\uplus\mathcal{F}],\Gamma_1)} M_1' \underset{\mathcal{L}}{\boxtimes} M_2'} \; \begin{array}{l} \mathcal{B} \subseteq \mathcal{E} \cap \mathcal{L} \;\wedge\; \neg(\exists M_2 \xrightarrow{(\mathcal{B}'[\mathcal{F}'],\Gamma_2')} M_2''. \\ (\mathcal{B}' \subseteq \mathcal{E} \cap \mathcal{L}) \wedge (|\mathcal{B}'| > |\mathcal{B}|)) \end{array}$$

Vertical Synchronisation Right

$$\frac{M_1 \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma_1)} M_1' \quad M_2 \xrightarrow{(\mathcal{B}[\mathcal{F}],\Gamma_2)} M_2'}{M_1 \underset{\mathcal{L}}{\boxtimes} M_2 \xrightarrow{(\mathcal{A}\cup\mathcal{B}[(\mathcal{F}\setminus\mathcal{A})\uplus\mathcal{E}],\Gamma_2)} M_1' \underset{\mathcal{L}}{\boxtimes} M_2'} \; \begin{array}{l} \mathcal{A} \subseteq \mathcal{F} \cap \mathcal{L} \;\wedge\; \neg(\exists M_1 \xrightarrow{(\mathcal{A}'[\mathcal{E}'],\Gamma_1')} M_1''. \\ (\mathcal{A}' \subseteq \mathcal{F} \cap \mathcal{L}) \wedge (|\mathcal{A}'| > |\mathcal{A}|)) \end{array}$$

Figure 5.5: Stochastic semantics of process algebra with hooks. Union of multi-sets is denoted by $\cup$, while sum of multi-sets is denoted by $\uplus$.

Definition 5.4 *Derivative Set.* The derivative set of a model process $M \in \mathbb{P}_m$ is denoted by $ds(M)$ and is defined as the smallest set of model processes such that:

- $M \in ds(M)$;
- if $M_i \in ds(M)$ and $M_i \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M_j$ then $M_j \in ds(M)$.

Definition 5.5 *Current moves of a definition process.* The multi set of moves that $D \in \mathbb{P}_d$ can perform is denoted by $Moves(D)$ and is defined as:

- $Moves(nil) = \{\!|\}$;
- $Moves(\mathcal{A}[\mathcal{E}].A) = \{\!|(\mathcal{A}[\mathcal{E}], A)|\!\}$;
- $Moves(D_1 + D_2) = Moves(D_1) \uplus Moves(D_2)$.

with $\{\!|\}$ delimiting a multi set and $\uplus$ the sum of multi sets.

Definition 5.6 *Current moves of a model process.* The multi set of moves that $M \in \mathbb{P}_m$ can perform is denoted by $Moves(M)$ and is defined as:

$((\mathcal{A}[\mathcal{E}], \Gamma), M') \in Moves(M)$ iff $M \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M'$, with the same multiplicity as the number of derivation trees that can derive $M \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M'$ using the derivation rules in Figure 5.5.

Definition 5.7 *Current composed actions for definition processes.* The multi set of composed actions that $D \in \mathbb{P}_d$ can perform is denoted by $CompAct(D)$ and is defined as:

$$CompAct(D) = \{\!|\mathcal{A}[\mathcal{E}] \mid (\mathcal{A}[\mathcal{E}], A) \in Moves(D)|\!\}$$

Definition 5.8 *Current activities for model Processes.* The multi set of activities that $M \in \mathbb{P}_m$ can perform is denoted by $Activities(M)$ and is defined as:

$$Activities(M) = \{\!|(\mathcal{A}[\mathcal{E}], \Gamma) \mid ((\mathcal{A}[\mathcal{E}], \Gamma), M') \in Moves(M)|\!\}$$

Definition 5.9 *Activity set.* The multi set of activities that a model process $M \in \mathbb{P}_m$ and its derivatives can perform is given by:

$$\overrightarrow{Activities}(M) = \biguplus_{M_i \in ds(M)} Activities(M_i)$$

Definition 5.10 *Derivation graph.* Given a model component $M \in \mathbb{P}_m$, the derivation graph $\mathcal{D}(M)$ is the labelled directed graph with:

- set of nodes $ds(M)$;
- multi set of transition labels $\overrightarrow{Activities}(M)$;
- multi set of labelled transitions $\rightarrow \subseteq ds(M) \times \overrightarrow{Activities}(M) \times ds(M)$. Given $M' \in ds(M)$, $(M', \mathcal{A}[\mathcal{E}], \Gamma, M'') \in \rightarrow$ iff $M' \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma)} M''$, with the same multiplicity of $((\mathcal{A}[\mathcal{E}], \Gamma), M'')$ in $Moves(M')$.

5.2.2 Rating sPAH models

In analogy with the functional rate evaluation in sSPA (Sections 3.2.3 and 3.2.4), we proceed now to the definition of a sPAH model, the rating routines and the rated derivation graph for sPAH models.

Definition 5.11 sPAH *model.* An sPAH model is a tuple:

$$(AgentDef, M, Actions, Names, \mathbb{F}, \Gamma, Participants, Var, Val)$$

where:

- $AgentDef$ is the finite set of agent definitions $\{A_1 \triangleq D_1,\ A_2 \triangleq D_2, \dots\}$;
- M is the initial state of the model, with $M \in \mathbb{P}_m$;
- $Actions$ is the finite set of actions;
- $Names$ is the finite set of parameter names;
- $\mathbb{F}$ is the finite set of functional rates;
- Γ is the finite set of constant model parameters, with $\Gamma \subseteq Names \times \mathbb{R}$;

- *Participants* is the finite set of sets of participants;
- *Var* and *Val* are the functions associating agents with variables (i.e. parameter names) and values, with $Var : \mathbb{P}_m \to Names$ and $Val : \mathbb{P}_m \to \mathbb{R}$.

In order to ensure correct and unambiguous rate evaluation and to guarantee that congruence relations (Chapter 6) can be defined on sPAH processes, we employ the following additional constraints:

- each functional rate $f_a \in \mathbb{F}$ is associated with a set of participants $p_a \subseteq Names$;
- at any time only one agent can be associated with a certain variable;
- whenever an agent A performs an action (application of derivation rule **Agent**), the resulting agent A' will be associated with the same variable A is associated with;
- only agents associated with variables in p_a can perform action a. This is to prevent an additional synchronisation via $\bowtie_{\mathcal{L}}$ changing a closed activity into an open activity, due to an increase of the size of Γ.
- actions used as hook actions must not be associated with functional rates;
- an activity $(\mathcal{A}[\mathcal{E}], \Gamma)$ can be rated only if Γ contains exactly the variables in p_a and $\mathcal{A}$ contains exactly one action name a such that $f_a \in \mathbb{F}$. Such activity is called *closed*. An activity that is not closed is called *open*;
- if more than one transition from a certain state is associated with the same functional rate, the evaluated rate has to be normalised, i.e. it has to be divided by the number of such transitions.

The above constraints are formalised by the following definitions. With these we can define a *rated derivation graph*.

Definition 5.12 *Well formed* sPAH *model.* A sPAH model is well formed if and only if:

1. Given a model process as a cooperation of agents of the form

$$A_1 \circ A_2 \circ \cdots \circ A_n$$

then $\forall A_i, A_j$ if $i \neq j$ then $Var(A_j) \neq Var(A_j)$, where $\circ$ is either a vertical or horizontal cooperation;

2. Given a definition of an agent A as a choice of sequential actions of the form

$$A \triangleq \sum_i a_i.A_i$$

then $\forall A_i \ Var(A) = Var(A_i)$;

3. $\forall a$ s.t. $f_a \in \mathbb{F}$, $\forall A$ agents

$$\exists (\mathcal{A}[\mathcal{E}], \Gamma) \in \overrightarrow{Activities}(A) \text{ s.t. } a \in \mathcal{A} \cup \mathcal{E} \Leftrightarrow Var(A) \in p_a$$

Moreover, whenever $M_1 \underset{\mathcal{L}}{\bowtie} M_2$ then $\forall a$ s.t. $f_a \in \mathbb{F}$

$$a \in \mathcal{L} \Leftrightarrow \exists (\mathcal{A}[\mathcal{E}], \Gamma) \in \overrightarrow{Activities}(M_1),$$
$$(\mathcal{B}[\mathcal{F}], \Gamma') \in \overrightarrow{Activities}(M_2) \text{ s.t. } a \in \mathcal{A} \wedge a \in \mathcal{B}$$

4. hook actions are not associated with functional rates:

$$\forall A \text{ agents}, \forall (\mathcal{A}[\mathcal{E}], \Gamma) \in \overrightarrow{Activities}(A), \ \forall a \text{ s.t. } f_a \in \mathbb{F}, \ a \notin \mathcal{E}$$

5. $\forall A$ agents defined as

$$A \triangleq \sum_i \mathcal{A}_i[\mathcal{H}_i].A_i$$

$\forall a$ s.t. $f_a \in \mathbb{F}$, if $a \in \mathcal{A}_i$ then $\mathcal{A}_i = \{\!|a|\!\}$.

Definition 5.12 ensures that whenever $M \xrightarrow{(\mathcal{A}[\mathcal{H}],\Gamma)} M'$ then either $\forall a$ s.t. $f_a \in \mathbb{F}$ $a \notin \mathcal{A}$ or $\exists! a$ s.t. $f_a \in \mathbb{F}$ and $a \in \mathcal{A}$. In other words, for every valid transition, the set of layer actions contains at most one action associated with a functional rate.

Definition 5.13 *Function* envVar. The function $envVar$ extracts the set of variables in an environment $\Gamma \subseteq Names \times \mathbb{R}$:

$$envVar(\Gamma) = (\{i \mid (i,k) \in \Gamma\})$$

Definition 5.14 *Function* activeActions. The function *activeActions* selects actions a such that a functional rate in the set $\mathbb{F}$ is associated with a, i.e. $f_a \in \mathbb{F}$, from a action set $A \subseteq Actions$:

$$activeActions(A)_{\mathbb{F}} = (\{a \mid a \in A \wedge f_a \in \mathbb{F}\})$$

Definition 5.15 *Open activity.* An open activity is an activity $(\mathcal{A}[\mathcal{E}], \Gamma)$ where at least one of the following conditions are true:

- the number of active actions in $\mathcal{A}$ is different from one, i.e. $|activeActions(\mathcal{A})_{\mathbb{F}}| \neq 1$;
- if $|activeActions(\mathcal{A})_{\mathbb{F}}| = 1$ and $a \in activeActions(\mathcal{A})_{\mathbb{F}}$, Γ does not contain the *exact* variables present in the participant set p_a, i.e. $p_a \neq envVar(\Gamma)$.

Definition 5.16 *Function* openActivities. The function *openActivities* selects open activities from a set of activities $A \subseteq 2^{Actions} \times 2^{Actions} \times 2^{Names \times \mathbb{R}}$:

$$openActivities(A) =$$
$$\left(\left\{ \left| (\mathcal{A}[\mathcal{E}], \Gamma) \;\middle|\; \begin{array}{c} (\mathcal{A}[\mathcal{E}], \Gamma) \in A \wedge (|activeActions(\mathcal{A})_{\mathbb{F}}| \neq 1 \\ \vee (activeActions(\mathcal{A})_{\mathbb{F}} = \{a\} \wedge p_a \neq envVar(\Gamma))) \end{array} \right| \right\} \right)$$

Definition 5.17 *Current open activities.* Given a model process $M \in \mathbb{P}_m$, the multi set of open activities that P can perform is defined as:

$$OpenAct(M) = openActivities(Activities(M))$$

Definition 5.18 *Open activity set.* The multi set of all open activities that a model process $M \in \mathbb{P}_m$ can perform is given by:

$$\overrightarrow{OpenAct}(M) = openActivities(\overrightarrow{Activities}(M))$$

Definition 5.19 *Closed activity.* A closed activity is an activity $(\mathcal{A}[\mathcal{E}], \Gamma)$ where:

- $|activeActions(\mathcal{A})_{\mathbb{F}}| = 1$, $a \in activeActions(\mathcal{A})_{\mathbb{F}}$ and Γ contains the exact variables present in the participant set p_a, i.e. $p_a = envVar(\Gamma)$.

Definition 5.20 *Function* closedActivities. The function *closedActivities* selects closed activities from a set of activities $A \subseteq 2^{Actions} \times 2^{Actions} \times 2^{Names \times \mathbb{R}}$:

$$closedActivities(A) = (A \setminus openActivities(A))$$

Definition 5.21 *Current closed activities.* Given a model process $M \in \mathbb{P}_m$, the multi set of closed activities that M can perform is defined as:

$$ClosedAct(M) = closedActivities(Activities(M))$$

Definition 5.22 *Closed activity set.* The multi set of all closed activities that a model process $M \in \mathbb{P}_m$ can perform is given by:

$$\overrightarrow{ClosedAct}(M) = closedActivities(\overrightarrow{Activities}(M))$$

Definition 5.23 *Open moves of a model process.* Given a model process $M \in \mathbb{P}_m$, the multi set of open moves of M, denoted $OpenMoves(M)$, is defined as:

$$OpenMoves(M) = \{\!|(a, M') \mid (a, M') \in Moves(M) \wedge a \in OpenAct(M)|\!\}$$

Definition 5.24 *Rated activity.* The pair $(\mathcal{A}[\mathcal{E}], r)$ such that $\mathcal{A}, \mathcal{E} \subseteq Actions$ and $r \in \mathbb{R}_{>0}$ is called a rated activity.

Definition 5.25 *Function* rateActivities. Given an environment $\Gamma \subseteq Names \times \mathbb{R}$, *rateActivities* converts a set of activities $A \subseteq 2^{Actions} \times 2^{Actions} \times 2^{Names \times \mathbb{R}}$ into a set of rated activities $B \subseteq 2^{Actions} \times 2^{Actions} \times \mathbb{R}$:

$$rateActivities(\Gamma)(A) =$$

$$\left\{ \left| (\mathcal{A}[\mathcal{E}], r) \;\middle|\; \begin{array}{l} (\mathcal{A}[\mathcal{E}], \Gamma') \in A \wedge \{a\} = activeActions(\mathcal{A})_{\mathbb{F}} \\ \wedge\ \Gamma \cup \Gamma' \vdash f_a \rightarrow k \wedge r_a = k/\pi(A, (a, \Gamma')) \wedge\ f_a \in \mathbb{F} \end{array} \right| \right\}$$

where $\pi(A, (a, \Gamma'))$ returns the number of occurrences of $(\mathcal{A}[\mathcal{E}], \Gamma')$ in the multi set A such that $\{a\} = activeActions(\mathcal{A})_{\mathbb{F}}$.

Definition 5.26 *Current rated activities.* Given a model process $M \in \mathbb{P}_m$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the multi set of rated activities that M can perform is defined as:

$$RatedAct(M)_\Gamma = rateActivities(\Gamma)(ClosedAct(M))$$

$RatedAct(M)_\Gamma$ can be written $RatedAct(M)$ if Γ is clear from the context.

Definition 5.27 *Rated activity set.* Given an environment $\Gamma \subseteq Names \times \mathbb{R}$, the multi set of rated activities that a model process $M \in \mathbb{P}_m$ and its derivatives can perform is given by:

$$\overrightarrow{RatedAct}(M)_\Gamma = rateActivities(\Gamma)(\overrightarrow{ClosedAct}(M))$$

$\overrightarrow{RatedAct}(M)_\Gamma$ can be written $\overrightarrow{RatedAct}(M)$ if Γ is clear from the context.

Definition 5.28 *Rated moves of a model process.* Given a model process $M \in \mathbb{P}_m$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the multi set of rated moves of M, denoted $RatedMoves(M)_\Gamma$, is defined as:

$$\begin{aligned} RatedMoves(M)_\Gamma = \{\!|((\mathcal{A}[\mathcal{E}], r), M') \mid ((\mathcal{A}[\mathcal{E}], \Gamma'), M') \in Moves(M') \wedge (\mathcal{A}[\mathcal{E}], \Gamma') \\ \in ClosedAct(M) \wedge \{\!|((\mathcal{A}[\mathcal{E}], k))|\!\} = rateActivities(\Gamma)(\{\!|(\mathcal{A}[\mathcal{E}], \Gamma')|\!\}) \wedge \\ activeActions(\mathcal{A})_{\mathbb{F}} = a \wedge r = k/\pi(ClosedAct(M), (a, \Gamma'))|\!\} \end{aligned}$$

Definition 5.29 *Rated transitions.* Given $M \in \mathbb{P}_m$ and $\Gamma \subseteq Names \times \mathbb{R}$, $M \xrightarrow{(\mathcal{A}[\mathcal{E}], r)}_\Gamma M'$ is a valid rated transition iff $M \xrightarrow{(\mathcal{A}[\mathcal{E}], \Gamma')} M'$, $(\mathcal{A}[\mathcal{E}], \Gamma') \in ClosedAct(M)$, $\{\!|((\mathcal{A}[\mathcal{E}], k))|\!\} = rateActivities(\Gamma)(\{\!|(\mathcal{A}[\mathcal{E}], \Gamma')|\!\})$, $activeActions(\mathcal{A})_{\mathbb{F}} = a$ and $r = k/\pi(ClosedAct(M), (a, \Gamma'))$.

Definition 5.30 *Rated derivation graph.* Given a model process $M \in \mathbb{P}_m$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the rated derivation graph $\mathcal{D}_r(M)_\Gamma$ is the labelled

directed graph with:

- set of nodes $ds(M)$;
- multi set of transition labels $\overrightarrow{RatedAct}(M)_\Gamma$;
- multi set of labelled transitions $\rightarrow_r \subseteq ds(M) \times \overrightarrow{RatedAct}(M)_\Gamma \times ds(M)$. Given $M' \in ds(M)$, $(M', \mathcal{A}[\mathcal{E}], r_a, M'') \in \rightarrow_r$ iff $M' \xrightarrow{(\mathcal{A}[\mathcal{E}], r_a)}_\Gamma M''$, with the same multiplicity of $((\mathcal{A}[\mathcal{E}], r_a), M'')$ in $RatedMoves(M')_\Gamma$.
- multi set of labelled transitions $\rightarrow_o \subseteq ds(M) \times \overrightarrow{OpenAct}(M) \times ds(M)$. Given $M' \in ds(M)$, $(M', \mathcal{A}[\mathcal{E}], \Gamma', M'') \in \rightarrow_o$ iff $M' \xrightarrow{(\mathcal{A}[\mathcal{E}], \Gamma')} M''$ and $(\mathcal{A}[\mathcal{E}], \Gamma') \in OpenAct(M)$, with the same multiplicity of $((\mathcal{A}[\mathcal{E}], \Gamma'), M'')$ in $Moves(M')$.

$\mathcal{D}_r(M)_\Gamma$ can be written $\mathcal{D}_r(M)$ if Γ is clear from the context.

In the following sections we illustrate the use of sPAH, augmenting examples of previous sections with functional rates.

5.2.3 Stochastic Process Algebra with Hooks and a Three Layers Example

In this section we show how the example in Section 5.1.2 can be augmented with functional rates. The processes are:

$$
\begin{array}{lll}
A_0 \triangleq a.A_1 + f.A_1 & B_0 \triangleq c.B_1 + e.B_1 & P_0 \triangleq p.P_1 \\
A_1 \triangleq a[p].A_2 + b.A_0 & B_1 \triangleq c[p].B_2 + d.B_0 & P_1 \triangleq q.P_0 + p[x].P_2 + \{\!|p, q|\!\}.P_1 \\
\quad + f[p].A_2 + e.A_0 & \quad + e[p].B_2 + f.B_0 & P_2 \triangleq q[y].P_1 \\
A_2 \triangleq b[q].A_1 + e[q].A_1 & B_2 \triangleq d[q].B_1 + f[q].B_1 & \\
Cell_M \triangleq move.Cell_M + x.Cell_A & Cell_A \triangleq absorb.Cell_A + y.Cell_M &
\end{array}
$$

The initial state of the model is given by:

$$((A_0 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} C_1$$

where $\mathcal{L} = \{\!|e, f|\!\}$, $\mathcal{H} = \{\!|p, q|\!\}$ and $\mathcal{K} = \{\!|x, y|\!\}$. Variables and values:

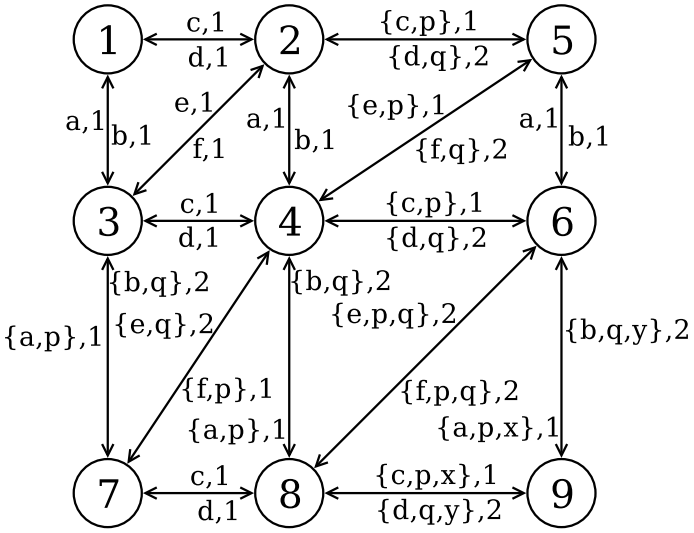


Figure 5.6: Rated derivation graph generated by the example in Section 5.2.3.

$$
\begin{array}{lll}
Var(A_0) = A & Var(A_1) = A & Var(A_2) = A \\
Val(A_0) = 0 & Val(A_1) = 1 & Val(A_2) = 2 \\
Var(B_0) = B & Var(B_1) = B & Var(B_2) = B \\
Val(B_0) = 0 & Val(B_1) = 1 & Val(B_2) = 2 \\
Var(P_0) = P & Var(P_1) = P & Var(P_2) = P \\
Val(P_0) = 0 & Val(P_1) = 1 & Val(P_2) = 2 \\
Var(Cell_M) = cell & Var(Cell_A) = cell & \\
Val(Cell_M) = 1 & Val(Cell_A) = 2 &
\end{array}
$$

The states of the model, reachable from the initial state, are:

$$
\begin{array}{ll}
1\text{: } ((A_0 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\boxtimes} P_0) \underset{\mathcal{K}}{\boxtimes} Cell_M & 2\text{: } ((A_0 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\boxtimes} P_0) \underset{\mathcal{K}}{\boxtimes} Cell_M \\
3\text{: } ((A_1 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\boxtimes} P_0) \underset{\mathcal{K}}{\boxtimes} Cell_M & 4\text{: } ((A_1 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\boxtimes} P_0) \underset{\mathcal{K}}{\boxtimes} Cell_M \\
5\text{: } ((A_0 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\boxtimes} P_1) \underset{\mathcal{K}}{\boxtimes} Cell_M & 6\text{: } ((A_1 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\boxtimes} P_1) \underset{\mathcal{K}}{\boxtimes} Cell_M \\
7\text{: } ((A_2 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\boxtimes} P_1) \underset{\mathcal{K}}{\boxtimes} Cell_M & 8\text{: } ((A_2 \underset{\mathcal{L}}{\bowtie} B_1) \underset{\mathcal{H}}{\boxtimes} P_1) \underset{\mathcal{K}}{\boxtimes} Cell_M \\
9\text{: } ((A_2 \underset{\mathcal{L}}{\bowtie} B_2) \underset{\mathcal{H}}{\boxtimes} P_2) \underset{\mathcal{K}}{\boxtimes} Cell_A &
\end{array}
$$

The functional rates and sets of participants associated to the actions are:

$$
\begin{aligned}
f_a &= k_a/h & p_a &= \{A\} \\
f_b &= (k_b * lA * h)/h & p_b &= \{A\} \\
f_c &= k_c/h & p_c &= \{B\} \\
f_d &= (k_d * lB * h)/h & p_d &= \{B\} \\
f_e &= (k_e * lA * h)/h & p_e &= \{A, B\} \\
f_f &= (k_f * lB * h)/h & p_f &= \{A, B\}
\end{aligned}
$$

A rated derivation graph can be produced using the additional environment set $\Gamma = \{(h,1),(k_a,1),(k_b,1),(k_c,1),(k_d,1),(k_e,1),(k_f,1)\}$. States and transitions of $\mathcal{D}_r(((A_0 \underset{\mathcal{L}}{\bowtie} B_0) \underset{\mathcal{H}}{\bowtie} P_0) \underset{\mathcal{K}}{\bowtie} Cell_M)_\Gamma$ are shown in Figure 5.6.

5.2.4 Stochastic Process Algebra with Hooks and Tissue Growth with Biochemistry

We show now how we can augment the example in Section 5.1.3 with functional rates. Agent definitions are as follows:

$$
\begin{aligned}
&NA0 \triangleq bioon0.A0_0 \\
&A0_0 \triangleq bioff0.NA0 + a0.A0_1 \\
&A0_1 \triangleq bioff0.NA0 + a0[mitoon0].A0_2 + b0.A0_0 \\
&A0_2 \triangleq bioff0.NA0 + b0[mitooff0].A0_1 \\
&\\
&Empty0 \triangleq mito10[bioon0].Tissueoff0 \\
&Tissueoff0 \triangleq apo0[bioff0].Empty0 + mitoon0.Tissueon0 \\
&Tissueon0 \triangleq apo0[bioff0].Empty0 + [mito01].Tissueon0 \\
&\qquad\qquad + mitooff0.Tissueoff0 \\
&\\
&NA1 \triangleq bioon1.A1_0 \\
&A1_0 \triangleq bioff1.NA1 + a1.A1_1 \\
&A1_1 \triangleq bioff1.NA1 + a1[mitoon1].A1_2 + b1.A1_0 \\
&A1_2 \triangleq bioff1.NA1 + b1[mitooff1].A1_1 \\
&\\
&Empty1 \triangleq mito21[bioon1].Tissueoff1
\end{aligned}
$$

$$+mito01[bioon1].Tissueoff1$$
$$Tissueoff1 \triangleq apo1[biooff1].Empty1 + mitoon1.Tissueon1$$
$$Tissueon1 \triangleq apo1[biooff1].Empty1 + mito12.Tissueon1$$
$$+mito10.Tissueon1 + mitooff1.Tissueoff1$$

$$NA2 \triangleq bioon2.A2_0$$
$$A2_0 \triangleq biooff2.NA2 + a2.A2_1$$
$$A2_1 \triangleq biooff2.NA2 + a2[mitoon2].A2_2 + b2.A2_0$$
$$A2_2 \triangleq biooff2.NA2 + b2[mitooff2].A2_1$$

$$Empty2 \triangleq mito12[bioon2].Tissueoff2$$
$$Tissueoff2 \triangleq apo2[biooff2].Empty2 + mitoon2.Tissueon2$$
$$Tissueon2 \triangleq apo2[biooff2].Empty2 + mito21.Tissueon2$$
$$+mitooff2.Tissueoff2$$

The initial state of the model is defined by the following process:

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\bowtie} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$

with $\mathcal{L} = \{\!|bioon0,\ biooff0,\ bioon1,\ biooff1,\ bioon2,\ biooff2,\ mitoon0,\ mitooff0,\ mitoon1,\ mitooff1, mitoon2, mitooff2|\!\}$, $\mathcal{K} = \{\!|mito10, mito01|\!\}$, $\mathcal{K} = \{\!|mito21, mito12|\!\}$.
Variables and values of the agents are:

$$Var(NA0) = A0 \quad Var(A0_0) = A0 \quad Var(A0_1) = A0 \quad Var(A0_2) = A0$$
$$Val(NA0) = 0 \quad Val(A0_0) = 0 \quad Val(A0_1) = 1 \quad Val(A0_2) = 2$$

$$Var(Empty0) = R0 \quad Var(Tissueoff0) = R0 \quad Var(Tissueon0) = R0$$
$$Val(Empty0) = 0 \quad Val(Tissueoff0) = 1 \quad Val(Tissueon0) = 2$$

$$Var(NA1) = A1 \quad Var(A1_0) = A1 \quad Var(A1_1) = A1 \quad Var(A1_2) = A1$$
$$Val(NA1) = 0 \quad Val(A1_0) = 0 \quad Val(A1_1) = 1 \quad Val(A1_2) = 2$$

$$Var(Empty1) = R1 \quad Var(Tissueoff1) = R1 \quad Var(Tissueon1) = R1$$
$$Val(Empty1) = 0 \quad Val(Tissueoff1) = 1 \quad Val(Tissueon1) = 2$$

$$Var(NA2) = A2 \quad Var(A2_0) = A2 \quad Var(A2_1) = A2 \quad Var(A2_2) = A2$$
$$Val(NA2) = 0 \quad Val(A2_0) = 0 \quad Val(A2_1) = 1 \quad Val(A2_2) = 2$$

$$Var(Empty2) = R2 \quad Var(Tissueoff2) = R2 \quad Var(Tissueon2) = R2$$
$$Val(Empty2) = 0 \quad Val(Tissueoff2) = 1 \quad Val(Tissueon2) = 2$$

Functional rates and sets of participants are:

$$f_{a0} = k_a/h \qquad p_{a0} = \{A0\}$$
$$f_{b0} = (k_b * A0 * h)/h \qquad p_{b0} = \{A0\}$$
$$f_{apo0} = k_{apo} \qquad p_{apo0} = \{R0\}$$

$$f_{a1} = k_a/h \qquad p_{a1} = \{A1\}$$
$$f_{b1} = (k_b * A1 * h)/h \qquad p_{b1} = \{A1\}$$
$$f_{apo1} = k_{apo} \qquad p_{apo1} = \{R1\}$$

$$f_{a2} = k_a/h \qquad p_{a2} = \{A2\}$$
$$f_{b2} = (k_b * A2 * h)/h \qquad p_{b2} = \{A2\}$$
$$f_{apo2} = k_{apo} \qquad p_{apo2} = \{R2\}$$

$$f_{mito01} = k_m \quad p_{mito01} = \{R0, R1\}$$
$$f_{mito12} = k_m \quad p_{mito12} = \{R1, R2\}$$
$$f_{mito10} = k_m \quad p_{mito10} = \{R1, R0\}$$
$$f_{mito21} = k_m \quad p_{mito21} = \{R2, R1\}$$

Examples of valid transitions are:

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\boxtimes} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$
$$\xrightarrow{(\{\!|b2, mitooff2|\!\},2)} (NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_1) \underset{\mathcal{L}}{\boxtimes} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueoff2)$$

$$(NA0 \underset{\emptyset}{\bowtie} A1_1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\boxtimes} (Empty0 \underset{\mathcal{K}}{\bowtie} Tissueoff1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$
$$\xrightarrow{(\{\!|apo1, biooff1|\!\},1)} (NA0 \underset{\emptyset}{\bowtie} NA1 \underset{\emptyset}{\bowtie} A2_2) \underset{\mathcal{L}}{\boxtimes} (Empty0 \underset{\mathcal{K}}{\bowtie} Empty1 \underset{\mathcal{K}'}{\bowtie} Tissueon2)$$

5.3 Summary

In this chapter we have introduced process algebra with hooks, a novel process algebra designed for multi-scale modelling of biological systems. The distinctive features of the algebra are *composed actions* and a new *vertical operator*. These new elements ensure that both the effects that an event produces at the scale where it originates and the effects that propagate to other scales can be modelled with a single transition. We illustrated the use of the algebra with several examples and we compared it with process algebra with priorities, illustrating how the former addresses the criticisms we highlighted in the latter. Finally, we developed a stochastic semantics for process algebra with hooks, which supports functional rates. This has been done following the approach to functional rates we developed in Chapter 3.

In the next chapter we introduce three fundamental equivalence relations for process algebra with hooks.

Chapter 6

Relations for Stochastic Process Algebra with Hooks

In this chapter we introduce three equivalence relations for stochastic process algebra with hooks: isomorphism ($\equiv$), $\mathcal{T}$-isomorphism ($\equiv_{\mathcal{T}}$) and Markovian $\mathcal{T}$-bisimulation ($\simeq_{\mathcal{T}}$). The last two equivalences relate processes at specified scales. In Section 6.1 we introduce the idea behind these relations and discuss the concept of the filtering of activities. In Section 6.2 we define the relations and give fundamental results such as equational laws and proofs of congruence. Finally, we discuss the practical use of these relations in Section 6.2.3.

6.1 Relating Biological Systems at Specified Scales

It is common practice to describe the behaviour of a biological system at a specific scale. When we do that, details of other scales are neglected as much as possible, only referred to when they are strictly necessary. For example, we compare the behaviour of two distinct populations of cells and observe that they proliferate at the same speed. The discussion is at the cellular scale, with cells as entities and the action of proliferation, or cell duplication, as event. Then, we might investigate further and discover that a cell duplicates in the first population only when a chemical **A** is present, in a certain amount. With this new observation, we are referring to a different scale: the molecular scale. Investigating even further, we might find that in the second population cell duplication is possible only if

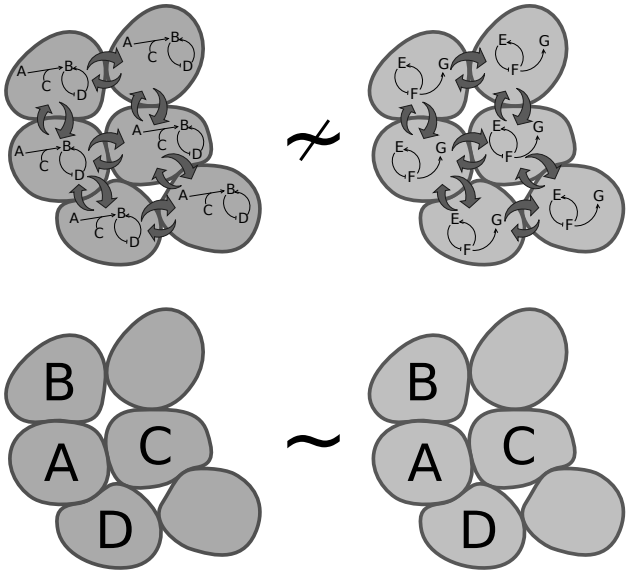


Figure 6.1: Two biological systems, if observed at different scales, can present distinct or analogous behaviour.

chemicals **E** and **F** are present, at the same time and in a certain quantity. Thus at the molecular scale the behaviour of the two populations is different. **Choosing which scale should be compared is responsibility of the modeller.** In this example, if we want to know whether the rate of growth of the two populations is the same, the answer is yes. If instead we want to know whether the molecular mechanisms that lead to cell duplications are the same, the answer is no. Figure 6.1 summarises this scenario: at the top of the figure, at the molecular scale, the cell populations are different, however, at the bottom of the figure, at the cellular scale, they are indistinguishable.

We will formalise these notions in sPAH with relations that will allow us to:

- compare the behaviour of two different process algebra models with respect to a specified scale;
- substitute parts of a model with behaviourally equivalent and less complex ones.

In order to relate models at a specified scale, we define a mechanism to focus on a specified scale in sPAH. This mechanism is called *filtering* and consists of removing undesired action names from valid rated transitions belonging to a rated derivation graph. The result will be a *filtered* derivation graph. Because actions

are associated with functional rates, removing actions interferes with rating. As a consequence, rating of transitions should always precede filtering.

In sPAH actions from every scale are collected into a unique action multi-set that labels valid transitions. If we want to focus on a specific scale, all we need to do is to keep only the actions pertaining to a given scale. For example, consider the following rated transition:

$$M \xrightarrow{(\{\!|a,h,x|\!\}[\{\!|y,z|\!\}],r)}_{\Gamma} M'$$

From this transition it is possible to infer that actions a, h and x have been performed, that hook actions y and z have not been used in any synchronisation and that the rate of the transition is r. Now assume that we are interested in only the behaviour represented by the actions in $\mathcal{T}$, with $\mathcal{T} = \{\!|x, y|\!\}$. The corresponding filtered transition should be:

$$M \xrightarrow{(\{\!|x|\!\},\{\!|y,z|\!\},r)}_{\mathcal{T},\Gamma} M'$$

Notice that the rate and the multi-set of hook actions is untouched. Moreover, a *filtered* activity is a triple, to distinguish it from a *rated* activity, which is a pair. We introduce now the formal definitions of filtered activities, filtered transitions and filtered derivation graph.

Definition 6.1 *Filtered activities.* The triple $(\mathcal{A}, \mathcal{E}, r)$ such that $\mathcal{A}$ and $\mathcal{E}$ are multi-sets of actions and $r \in \mathbb{R}_{>0}$ is called a filtered activity.

Definition 6.2 *Function* filterActivities. Given a multi-set of actions $\mathcal{T}$, *filterActivities* converts a multi-set of rated activities A into a multi-set of filtered activities:

$$filterActivities(\mathcal{T})(A) = \{\!|(\mathcal{B}, \mathcal{E}, r) \mid (\mathcal{A}, \mathcal{E}, r) \in A \wedge \mathcal{B} = \mathcal{A} \cap \mathcal{T}|\!\}$$

Definition 6.3 *Current filtered composed actions of a definition process.* Given a multi-set of actions $\mathcal{T}$, the multi-set of filtered composed actions that $D \in \mathbb{P}_d$ can perform is denoted by $FiltCompAct(D)_{\mathcal{T}}$ and is defined as:

$$FiltCompAct(D)_{\mathcal{T}} = \{\!|(\mathcal{B}, \mathcal{E}) \mid \mathcal{A}[\mathcal{E}] \in CompAct(D) \wedge \mathcal{B} = \mathcal{A} \cap \mathcal{T}|\!\}$$

Definition 6.4 *Current filtered activities of a model process.* Given a multi-set of actions $\mathcal{T}$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the multi-set of filtered activities that a model process $M \in \mathbb{P}_m$ can perform is denoted by $FiltAct(M)_{\mathcal{T},\Gamma}$ and is defined as:

$$FiltAct(M)_{\mathcal{T},\Gamma} = filterActivities(\mathcal{T})(RatedAct(M)_{\Gamma})$$

Definition 6.5 *Filtered activity set.* Given a multi-set of actions $\mathcal{T}$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the multi-set of filtered activities that a model process $M \in \mathbb{P}_m$ and its derivatives can perform is denoted by $\overrightarrow{FiltAct}(M)_{\mathcal{T},\Gamma}$ and is defined as:

$$\overrightarrow{FiltAct}(M)_{\mathcal{T},\Gamma} = filterActivities(\mathcal{T})(\overrightarrow{RatedAct(M)}_{\Gamma})$$

Definition 6.6 *Filtered moves of a definition process.* Given a multi-set of actions $\mathcal{T}$, the multi-set of moves that a definition process $D \in \mathbb{P}_d$ can perform is denoted by $FiltMovesD_{\mathcal{T}}$ and is defined as:

$$FiltMoves(D)_{\mathcal{T}} = \{\!|((\mathcal{B},\mathcal{E}),A) \mid (\mathcal{A}[\mathcal{E}],A) \in Moves(D) \wedge \mathcal{B} = \mathcal{A} \cap \mathcal{T}|\!\}$$

Definition 6.7 *Filtered moves of a model process.* Given a multi-set of actions $\mathcal{T}$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the multi-set of moves that a model process $M \in \mathbb{P}_m$ can perform is denoted by $FiltMovesM_{\mathcal{T},\Gamma}$ and is defined as:

$$FiltMoves(M)_{\mathcal{T},\Gamma} = \{\!|((\mathcal{B},\mathcal{E},r),A) \mid ((\mathcal{A}[\mathcal{E}],r),A) \in RatedMoves(M) \wedge \mathcal{B} = \mathcal{A} \cap \mathcal{T}|\!\}$$

Definition 6.8 *Filtered transitions of a definition process.* Given $D \in \mathbb{P}_d$ and a multi-set of actions $\mathcal{T}$, $D \xrightarrow{(\mathcal{B},\mathcal{E})}_{\mathcal{T}} A$ is a valid filtered transition iff $D \xrightarrow{\mathcal{A}[\mathcal{E}]} A$ for some $\mathcal{A}$ such that $\mathcal{B} = \mathcal{A} \cap \mathcal{T}$.

Definition 6.9 *Filtered transitions of a model process.* Given $M \in \mathbb{P}_m$, a multi-set of actions $\mathcal{T}$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, $M \xrightarrow{(\mathcal{B},\mathcal{E},r)}_{\mathcal{T},\Gamma} M'$ is a valid filtered transition iff $M \xrightarrow{(\mathcal{A}[\mathcal{E}],r)}_{\Gamma} M'$ for some $\mathcal{A}$ such that $\mathcal{B} = \mathcal{A} \cap \mathcal{T}$.

Definition 6.10 *Filtered derivation graph.* Given a model process $M \in \mathbb{P}_m$, a multi-set of actions $\mathcal{T}$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, the filtered derivation graph $\mathcal{D}_f(M)_{\mathcal{T},\Gamma}$ is the labelled directed graph with:

- set of nodes $ds(M)$;
- multi-set of transition labels $\overrightarrow{FiltAct}(M)_{\mathcal{T},\Gamma}$;
- multi-set of labelled transitions $\rightarrow_f \subseteq ds(M) \times \overrightarrow{FiltAct}(M)_{\mathcal{T},\Gamma} \times ds(M)$. Given $M' \in ds(M)$, $(M', \mathcal{A}, \mathcal{E}, r_a, M'') \in \rightarrow_f$ iff $M' \xrightarrow{(\mathcal{A},\mathcal{E},r_a)}_{\mathcal{T},\Gamma} M''$, with the same multiplicity of $((\mathcal{A}, \mathcal{E}, r_a), M'')$ in $FiltMoves(M')_{\mathcal{T},\Gamma}$.
- multi-set of labelled transitions $\rightarrow_o \subseteq ds(M) \times \overrightarrow{OpenAct}(M) \times ds(M)$. Given $M' \in ds(M)$, $(M', \mathcal{A}[\mathcal{E}], \Gamma', M'') \in \rightarrow_o$ iff $M' \xrightarrow{(\mathcal{A}[\mathcal{E}],\Gamma')} M''$ and $(\mathcal{A}[\mathcal{E}], \Gamma') \in OpenAct(M)$, with the same multiplicity of $((\mathcal{A}[\mathcal{E}], \Gamma'), M'')$ in $Moves(M')$.

$\mathcal{D}_f(M)_{\mathcal{T},\Gamma}$ can be written $\mathcal{D}_f(M)$ if $\mathcal{T}$ and Γ are clear from the context.

With the notion of filtered derivation graph we can now define relations between sPAH processes based on filtering and relate sPAH processes at specified scales.

6.2 Three Fundamental Relations

Three equivalence relations on sPAH processes are defined: *isomorphism* ($\equiv$), $\mathcal{T}$*-isomorphism* ($\equiv_{\mathcal{T}}$) and *Markovian* $\mathcal{T}$*-bisimulation* ($\simeq_{\mathcal{T}}$). In particular:

- *Isomorphism* ($\equiv$) ensures that two processes generate equivalent derivation graphs. This equivalence is used to prove fundamental equivalence laws, such as $P_1 \underset{\mathcal{L}}{\bowtie} P_2 \equiv P_2 \underset{\mathcal{L}}{\bowtie} P_1$;
- $\mathcal{T}$*-isomorphism* ($\equiv_{\mathcal{T}}$) ensures that two processes generate equivalent derivation graphs after rating and filtering activities;
- *Markovian* $\mathcal{T}$*-bisimulation* ($\simeq_{\mathcal{T}}$) ensures that two processes produce the same rated and filtered activities at the same time and with the same probability, while presenting identical open transitions.

Before introducing the formal definitions of three fundamental relations, we illustrate them with an example. Consider the following processes:

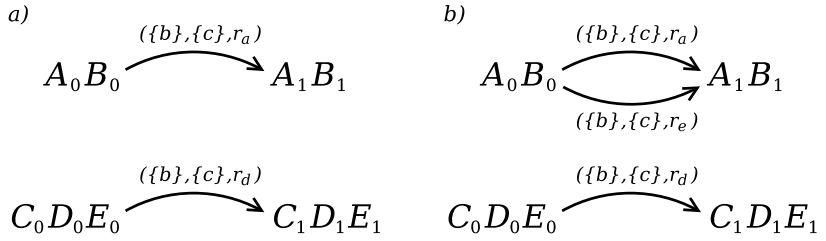


Figure 6.2: Filtered derivation graphs of the example in this section. *a*) if $r_a = r_d$, the transition systems are $\mathfrak{T}$-isomorphic. *b*) if $r_d = r_a + r_e = r$ the transition systems are Markovian $\mathfrak{T}$-bisimilar.

$$\begin{array}{llll} A_0 \triangleq a[b].A_1 & D_0 \triangleq b[c].D_1 & A_1 \triangleq nil & D_1 \triangleq nil \\ B_0 \triangleq b[c].B_1 & E_0 \triangleq d.E_1 & B_1 \triangleq nil & E_1 \triangleq nil \\ C_0 \triangleq d[b].C_1 & & C_1 \triangleq nil & \end{array}$$

Consider two sPAH model processes: $A_0 \underset{\{\!|b|\!\}}{\bowtie} B_0$ and $(C_0 \underset{\{\!|b|\!\}}{\bowtie} D_0) \underset{\{\!|d|\!\}}{\bowtie} E_0$. The following transitions are possible:

$$A_0 \underset{\{\!|b|\!\}}{\bowtie} B_0 \xrightarrow{(\{\!|a,b|\!\}[c],\Gamma)} A_1 \underset{\{\!|b|\!\}}{\bowtie} B_1$$
$$(C_0 \underset{\{\!|b|\!\}}{\bowtie} D_0) \underset{\{\!|d|\!\}}{\bowtie} E_0 \xrightarrow{(\{\!|d,b|\!\}[c],\Gamma')} (C_1 \underset{\{\!|b|\!\}}{\bowtie} D_1) \underset{\{\!|d|\!\}}{\bowtie} E_1$$

These are the only possible transitions. We cannot consider them equivalent in the sense that they generate isomorphic derivation graphs, so $A_0 \underset{\{\!|b|\!\}}{\bowtie} B_0 \not\equiv (C_0 \underset{\{\!|b|\!\}}{\bowtie} D_0) \underset{\{\!|d|\!\}}{\bowtie} E_0$. However, if we decide to select only actions in the multi-set $\mathfrak{T} = \{\!|b|\!\}$, and rating yields rates r_a and r_d, we obtain transitions:

$$A_0 \underset{\{\!|b|\!\}}{\bowtie} B_0 \xrightarrow{(\{\!|b|\!\},\{\!|c|\!\},r_a)}_{\mathfrak{T}} A_1 \underset{\{\!|b|\!\}}{\bowtie} B_1$$
$$(C_0 \underset{\{\!|b|\!\}}{\bowtie} D_0) \underset{\{\!|d|\!\}}{\bowtie} E_0 \xrightarrow{(\{\!|b|\!\},\{\!|c|\!\},r_d)}_{\mathfrak{T}} (C_1 \underset{\{\!|b|\!\}}{\bowtie} D_1) \underset{\{\!|d|\!\}}{\bowtie} E_1$$

If $r_a = r_d$, the two models generate $\mathfrak{T}$-isomorphic filtered derivation graphs, written $A_0 \underset{\{\!|b|\!\}}{\bowtie} B_0^l \equiv_{\{\!|b|\!\}} (C_0 \underset{\{\!|b|\!\}}{\bowtie} D_0) \underset{\{\!|d|\!\}}{\bowtie} E_0$ (Figure 6.2, *a*). We will show later that $\equiv_{\mathfrak{T}}$ is a congruence for sPAH processes, which means that substituting one for the other within a larger sPAH model will produce a sPAH model that is $\mathfrak{T}$-isomorphic to the original one.

Now assume $A_0 \triangleq a[b].A_1 + e[b].A_1$, where e is a biochemical action. This produces an additional transition for $A_0 \underset{\{\!|b|\!\}}{\bowtie} B_0$:

$$A_0 \underset{\{\!|b|\!\}}{\bowtie} B_0 \xrightarrow{(\{\!|e,b|\!\}[c],\Gamma'')} A_1 \underset{\{\!|b|\!\}}{\bowtie} B_1$$

that filtered for $\{\!|b|\!\}$ becomes:

$$A_0 \underset{\{\!|b|\!\}}{\boxtimes} B_0 \xrightarrow{(\{\!|b|\!\},\{\!|c|\!\},r_e)}_{\mathcal{T}} A_1 \underset{\{\!|b|\!\}}{\boxtimes} B_1$$

As a result the two models are no longer $\mathcal{T}$-isomorphic. However, if $r_d = r_a + r_e = r$, both $A_0 \underset{b}{\boxtimes} B_0$ and $(C_0 \underset{\{\!|b|\!\}}{\boxtimes} D_0) \underset{\{\!|d|\!\}}{\bowtie} E_0$ can move to a terminal state with filtered set of layer actions $\{\!|b|\!\}$ and set of hook actions $\{\!|c|\!\}$ with a total rate of r. In other words, the pair $(\{\!|b|\!\}, \{\!|c|\!\})$ appears on an activity at the same time with the same probability, implying the two model processes are Markovian $\mathcal{T}$-bisimilar, written $A_0 \underset{\{\!|b|\!\}}{\boxtimes} B_0 \simeq_{\{\!|b|\!\}} (C_0 \underset{\{\!|b|\!\}}{\boxtimes} D_0) \underset{\{\!|d|\!\}}{\bowtie} E_0$ (Figure 6.2, *b*). Again, we will demonstrate $\simeq_{\mathcal{T}}$ is a congruence for process algebra with hooks processes.

In the next two sections we define formally the three equivalence relations and their fundamental properties.

6.2.1 Isomorphism and $(\mathcal{T}, \Gamma)$-isomorphism

In this section we define formally isomorphism ($\equiv$) and $(\mathcal{T}, \Gamma)$-isomorphism ($\equiv_{\mathcal{T},\Gamma}$) on sPAH processes. Two sPAH processes are considered isomorphic if their derivation graphs are equivalent, i.e. if it is possible to define a bijection $\mathcal{F}$ between the states of the two derivation graphs such that the moves of the image under $\mathcal{F}$ of a state are always identical to the moves of the same states where in each move the resulting state is replaced by its image under $\mathcal{F}$. A similar definition is given for $(\mathcal{T}, \Gamma)$-isomorphism, where a filtered derivation graph is considered instead of a derivation graph.

In addition we prove that $\equiv \subset \equiv_{\mathcal{T},\Gamma}$ (Propositions 6.18 and 6.19), we give equational laws for $\equiv$ (Proposition 6.20) and we show that both $\equiv$ and $\equiv_{\mathcal{T},\Gamma}$ are congruences (Propositions 6.22 and 6.23).

Definition 6.11 *Function* apply. Given a function $f : \mathbb{P} \to \mathbb{P}$ and a multi-set of filtered moves $MSet$, function *apply* applies f to $MSet$ in the following way:

$$apply(f)(MSet) = \{\!|(a, f(P)) \mid (a, P) \in MSet|\!\}$$

Definition 6.12 *Model process isomorphism.* A function $\mathcal{F}$: $ds(M_1) \to ds(M_2)$ is a model process isomorphism between M_1 and M_2 $(M_1, M_2 \in \mathbb{P}_m)$, if $\mathcal{F}$ is

bijective and $\forall M_1' \in ds(M_1)$,

$$Moves(\mathcal{F}(M_1')) = apply(\mathcal{F})(Moves(M_1'))$$

Definition 6.13 *Isomorphic model processes.* Two model processes $M_1, M_2 \in \mathbb{P}_m$ are isomorphic, written $M_1 \equiv M_2$, if there is a model process isomorphism $\mathcal{F}$ between them such that $\mathcal{D}(\mathcal{F}(M_1)) = \mathcal{D}(M_2)$.

Definition 6.14 *Model process* $(\mathcal{T}, \Gamma)$*-isomorphism.* Given an environment $\Gamma \subseteq Names \times \mathbb{R}$ and an action multi-set $\mathcal{T}$, a function $\mathcal{F}$: $ds(M_1) \rightarrow ds(M_2)$ is a model component $(\mathcal{T}, \Gamma)$-isomorphism between M_1 and M_2 ($M_1, M_2 \in \mathbb{P}_m$), if $\mathcal{F}$ is bijective and $\forall M_1' \in ds(M_1)$,

$$FiltMoves(\mathcal{F}(M_1'))_{\mathcal{T},\Gamma} = apply(\mathcal{F})(FiltMoves(M_1')_{\mathcal{T},\Gamma})$$

and

$$OpenMoves(\mathcal{F}(M_1')) = apply(\mathcal{F})(OpenMoves(M_1'))$$

Definition 6.15 $(\mathcal{T}, \Gamma)$*-isomorphic model processes.* Given an environment $\Gamma \subseteq Names \times \mathbb{R}$ and an action multi-set $\mathcal{T}$, two model processes $M_1, M_2 \in \mathbb{P}_m$ are $(\mathcal{T}, \Gamma)$-isomorphic, written $M_1 \equiv_{\mathcal{T},\Gamma} M_2$, if there is a model component $(\mathcal{T}, \Gamma)$-isomorphism $\mathcal{F}$ between them such that $\mathcal{D}_f(\mathcal{F}(M_1))_{\mathcal{T},\Gamma} = \mathcal{D}_f(M_2)_{\mathcal{T},\Gamma}$.

If Γ is clear from the context, we write $M_1 \equiv_{\mathcal{T}} M_2$ instead of $M_1 \equiv_{\mathcal{T},\Gamma} M_2$ and we say M_1 and M_2 are $\mathcal{T}$-isomorphic.

Definition 6.16 *Isomorphic definition processes.* Two definition processes $D_1, D_2 \in \mathbb{P}_d$ are isomorphic ($D_1 \equiv D_2$) iff there exists an bijective function $\mathcal{F} : ds(D_1) \rightarrow ds(D_2)$ such that $\forall A \in ds(D_1)$, $A \equiv \mathcal{F}(A)$ and

$$Moves(D_2) = apply(\mathcal{F})(Moves(D_1))$$

Definition 6.17 $(\mathcal{T}, \Gamma)$*-isomorphic definition processes.* Given a multi-set of actions $\mathcal{T}$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, two definition processes $D_1, D_2 \in \mathbb{P}_d$ are $(\mathcal{T}, \Gamma)$-isomorphic ($D_1 \equiv_{\mathcal{T},\Gamma} D_2$) iff there exists an bijective function $\mathcal{F}$:

$ds(D_1) \to ds(D_2)$ such that $\forall A \in ds(D_1)$, $A \equiv_{\mathcal{T},\Gamma} \mathcal{F}(A)$ and

$$FiltMoves(D_2)_{\mathcal{T}} = apply(\mathcal{F})(FiltMoves(D_1)_{\mathcal{T}})$$

Proposition 6.18 Given $M_1, M_2 \in \mathbb{P}_m$,

$$M_1 \equiv M_2 \Rightarrow \forall \text{ multi-sets of actions } \mathcal{T}, \forall \Gamma \subseteq Names \times \mathbb{R},\ M_1 \equiv_{\mathcal{T},\Gamma} M_2$$

Proof. We just need to prove is that $\forall$ multi-sets of actions $\mathcal{T}$, $\forall \Gamma \subseteq Names \times \mathbb{R}$, $\forall M_1' \in ds(M_1)$ if $Moves(\mathcal{F}(M_1')) = apply(\mathcal{F})(Moves(M_1'))$ then

$$FiltMoves(\mathcal{F}(M_1'))_{\mathcal{T},\Gamma} = apply(\mathcal{F})(FiltMoves(M_1')_{\mathcal{T},\Gamma})$$

and

$$OpenMoves(\mathcal{F}(M_1')) = apply(\mathcal{F})(OpenMoves(M_1')$$

which is trivially true because filtered moves and open moves are derived from the moves of processes in the same way.

□

Proposition 6.19 Given $D_1, D_2 \in \mathbb{P}_d$,

$$D_1 \equiv D_2 \Rightarrow \forall \text{ multi-sets of actions } \mathcal{T}, \forall \Gamma \subseteq Names \times \mathbb{R},\ D_1 \equiv_{\mathcal{T},\Gamma} D_2$$

Proof. Observe that if $D_1 \equiv D_2$ then there exists an bijective function $\mathcal{F} : ds(D_1) \to ds(D_2)$ such that $\forall A \in ds(D_1)$ then $A \equiv \mathcal{F}(A)$ and

$$Moves(D_2) = apply(\mathcal{F})(Moves(D_1))$$

This implies that $\forall$ multi-sets of actions $\mathcal{T}$, $\forall \Gamma \subseteq Names \times \mathbb{R}$, $\forall A \in ds(D_1)$ then $A \equiv_{\mathcal{T},\Gamma} \mathcal{F}(A)$ because of Proposition 6.18. Moreover, $FiltMoves(D_2)_{\mathcal{T}} = apply(\mathcal{F})(FiltMoves(D_1)_{\mathcal{T}})$ because filtered moves are derived in the same way from the moves of processes.

□

Proposition 6.20 *Equational laws for isomorphic* sPAH *processes.* The following laws can be proved using the stochastic operational semantics and the definition of model and definition process isomorphisms. These are only some examples of equational laws that can be proved on sPAH processes.

1. $D_1 + D_2 \equiv D_2 + D_1$;
2. $(D_1 + D_2) + D_3 \equiv D_1 + (D_2 + D_3)$;
3. $M_1 \underset{\mathcal{L}}{\bowtie} M_2 \equiv M_2 \underset{\mathcal{L}}{\bowtie} M_1$;
4. $(M_1 \underset{\mathcal{L}}{\bowtie} M_2) \underset{\mathcal{L}}{\bowtie} M_3 \equiv M_1 \underset{\mathcal{L}}{\bowtie} (M_2 \underset{\mathcal{L}}{\bowtie} M_3)$;
5. $M_1 \underset{\mathcal{L}}{\boxtimes} M_2 \equiv M_2 \underset{\mathcal{L}}{\boxtimes} M_1$;
6. $(M_1 \underset{\mathcal{L}}{\bowtie} M_2) \underset{\mathcal{K}}{\bowtie} M_3 \equiv M_1 \underset{\mathcal{L}}{\bowtie} (M_2 \underset{\mathcal{K}}{\bowtie} M_3)$, if $\forall(\mathcal{A}[\mathcal{E}], \Gamma) \in \overrightarrow{Activities}(M_1)$, $\forall(\mathcal{N}[\mathcal{H}], \Gamma'') \in \overrightarrow{Activities}(M_3)$, $\mathcal{N} \cap (\mathcal{L} \setminus \mathcal{K}) = \emptyset \wedge \mathcal{A} \cap (\mathcal{K} \setminus \mathcal{L}) = \emptyset$;

Proof. We prove each law in turn:

1. $Moves(D_1 + D_2) = Moves(D_1) \uplus Moves(D_2) = Moves(D_2 + D_1)$ with $\mathcal{F}$ the identity function $id : \mathbb{P}_m \to \mathbb{P}_m$.
2. proof analogous to 1.
3. We choose model process isomorphism $\mathcal{F}$ as

$$\forall M_1' \underset{\mathcal{L}}{\bowtie} M_2' \in ds(M_1 \underset{\mathcal{L}}{\bowtie} M_2),\ \mathcal{F}(M_1' \underset{\mathcal{L}}{\bowtie} M_2') = M_2' \underset{\mathcal{L}}{\bowtie} M_1'$$

   with $M_1' \in ds(M_1)$ and $M_2' \in ds(M_2)$. Clearly, because of the symmetry of operator $\underset{\mathcal{L}}{\bowtie}$,

$$Moves(\mathcal{F}(M_1' \underset{\mathcal{L}}{\bowtie} M_2')) = apply(\mathcal{F})(Moves(M_1' \underset{\mathcal{L}}{\bowtie} M_2'))$$

4. proof analogous to 3.
5. We choose model process isomorphism $\mathcal{F}$ as

$$\forall M_1' \underset{\mathcal{L}}{\boxtimes} M_2' \in ds(M_1 \underset{\mathcal{L}}{\boxtimes} M_2),\ \mathcal{F}(M_1' \underset{\mathcal{L}}{\boxtimes} M_2') = M_2' \underset{\mathcal{L}}{\boxtimes} M_1'$$

with $M_1' \in ds(M_1)$ and $M_2' \in ds(M_2)$. Clearly, because of the symmetry of operator $\underset{\mathcal{L}}{\bowtie}$,

$$Moves(\mathcal{F}(M_1' \underset{\mathcal{L}}{\bowtie} M_2')) = apply(\mathcal{F})(Moves(M_1' \underset{\mathcal{L}}{\bowtie} M_2'))$$

6. We choose model process isomorphism $\mathcal{F}$ as

$$\forall (M_1' \underset{\mathcal{L}}{\bowtie} M_2') \underset{\mathcal{K}}{\bowtie} M_3' \in ds((M_1 \underset{\mathcal{L}}{\bowtie} M_2) \underset{\mathcal{K}}{\bowtie} M_3),$$
$$\mathcal{F}((M_1' \underset{\mathcal{L}}{\bowtie} M_2') \underset{\mathcal{K}}{\bowtie} M_3') = M_1' \underset{\mathcal{L}}{\bowtie} (M_2' \underset{\mathcal{K}}{\bowtie} M_3')$$

with $M_1' \in ds(M_1)$, $M_2' \in ds(M_2)$ and $M_3' \in ds(M_3)$. Using the additional conditions of 6. we have

$$Moves(\mathcal{F}((M_1' \underset{\mathcal{L}}{\bowtie} M_2') \underset{\mathcal{K}}{\bowtie} M_3')) = apply(\mathcal{F})((Moves(M_1' \underset{\mathcal{L}}{\bowtie} M_2') \underset{\mathcal{K}}{\bowtie} M_3'))$$

□

Proposition 6.21 *Equational laws for* $(\mathcal{T}, \Gamma)$*-isomorphic* sPAH *processes.* Because of Propositions 6.18 and 6.19, the equational laws for isomorphic sPAH processes hold as equational laws for $(\mathcal{T}, \Gamma)$-isomorphic sPAH processes.

Proposition 6.22 *Isomorphism as a Congruence.* If $P_1, P_2 \in \mathbb{P}$ such that $P_1 \equiv P_2$, then

1. $\mathcal{A}[\mathcal{E}].P_1 \equiv \mathcal{A}[\mathcal{E}].P_2$, with P_1, P_2 agents
2. $P_1 + Q \equiv P_2 + Q$, with $P_1, P_2, Q \in \mathbb{P}_d$
3. $P_1 \underset{\mathcal{L}}{\bowtie} Q \equiv P_2 \underset{\mathcal{L}}{\bowtie} Q$, with $P_1, P_2, Q \in \mathbb{P}_m$
4. $P_1 \underset{\mathcal{L}}{\bowtie} Q \equiv P_2 \underset{\mathcal{L}}{\bowtie} Q$, with $P_1, P_2, Q \in \mathbb{P}_m$

Proof. We prove each case in turn:

1. $\mathcal{A}[\mathcal{E}].P_1$ and $\mathcal{A}[\mathcal{E}].P_2$ are definition processes. Because $P_1 \equiv P_2$ there exists model process isomorphism $\mathcal{F}$ between them such that $\mathcal{F}(P_1) = P_2$ and

$P_1 \equiv \mathcal{F}(P_1)$. Clearly, we have

$$Moves(\mathcal{A}[\mathcal{E}].P_2) = apply(\mathcal{F})(Moves(\mathcal{A}[\mathcal{E}].P_1))$$

2. From the assumptions we know that $\exists \mathcal{F} : ds(P_1) \to ds(P_2)$ bijective such that $\mathcal{F}(P_1) = P_2$ and $\forall A \in ds(P_1)$, $A \equiv \mathcal{F}(A)$ and

$$Moves(P_2) = apply(\mathcal{F})(Moves(P_1))$$

Thus, we need $\mathcal{G} : ds(P_1 + Q) \to ds(P_2 + Q)$ bijective such that $\forall A \in ds(P_1 + Q)$, $A \equiv \mathcal{F}(A)$ and

$$Moves(P_2 + Q) = apply(\mathcal{G})(Moves(P_1 + Q))$$

We define $\mathcal{G}$ as:

$$\mathcal{G}(A) = \begin{cases} A, & \text{if } Q \xrightarrow{a} A \\ \mathcal{F}(A), & \text{if } P_1 \xrightarrow{a} A \end{cases}$$

Both cases of $\mathcal{G}$ ensure that $G(A) \equiv A$. Finally:

$$Moves(P_2 + Q) = Moves(P_2) \uplus Moves(Q) =$$

$$apply(\mathcal{F})(Moves(P_1)) \uplus apply(id)(Moves(Q)) = apply(\mathcal{G})(Moves(P_1 + Q))$$

3. We know there is a model process isomorphism $\mathcal{F}$ between P_1 and P_2. Each element of $ds(P_1 \underset{\mathcal{L}}{\bowtie} Q)$ has the form $P_1' \underset{\mathcal{L}}{\bowtie} Q'$. We define a model process isomorphism $\mathcal{G}$ as: $\forall P_1' \underset{\mathcal{L}}{\bowtie} Q' \in ds(P_1 \underset{\mathcal{L}}{\bowtie} Q)$, with $P_1' \in ds(P_1)$ and $Q' \in ds(Q)$,

$$\mathcal{G}(P_1' \underset{\mathcal{L}}{\bowtie} Q') = \mathcal{F}(P_1') \underset{\mathcal{L}}{\bowtie} Q'$$

$\mathcal{G}$ is a model process isomorphism because $\mathcal{F}$ is a model process isomorphism.

In fact:

$$
\begin{aligned}
&Moves(\mathcal{G}(P_1' \underset{\mathcal{L}}{\bowtie} Q')) = Moves(\mathcal{F}(P_1') \underset{\mathcal{L}}{\bowtie} Q') = \\
&\quad \{\!|((\mathcal{A}[\mathcal{E}], \Gamma), R \underset{\mathcal{L}}{\bowtie} Q') \mid ((\mathcal{A}[\mathcal{E}], \Gamma), R) \in Moves(\mathcal{F}(P_1')) \wedge \mathcal{A} \cap \mathcal{L} = \emptyset|\!\} \uplus \\
&\quad \{\!|((\mathcal{A}[\mathcal{E}], \Gamma), \mathcal{F}(P_1') \underset{\mathcal{L}}{\bowtie} Q'') \mid ((\mathcal{A}[\mathcal{E}], \Gamma), Q'') \in Moves(Q') \wedge \mathcal{A} \cap \mathcal{L} = \emptyset|\!\} \uplus \\
&\quad \{\!|((\mathcal{A}[\mathcal{E}], \Gamma), R \underset{\mathcal{L}}{\bowtie} Q'')) \mid ((\mathcal{A}_1[\mathcal{E}_1], \Gamma_1), R) \in Moves(\mathcal{F}(P_1')) \wedge \\
&\quad ((\mathcal{A}_2[\mathcal{E}_2], \Gamma_2), Q'') \in Moves(Q') \wedge \mathcal{A}_1 \cap \mathcal{A}_2 \cap \mathcal{L} \neq \emptyset \wedge \\
&\quad \mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \wedge \mathcal{E} = \mathcal{E}_1 \uplus \mathcal{E}_2 \wedge \Gamma = \Gamma_1 \cup \Gamma_2|\!\} = \\
&\quad \{\!|((\mathcal{A}[\mathcal{E}], \Gamma), \mathcal{F}(P_1'') \underset{\mathcal{L}}{\bowtie} Q') \mid ((\mathcal{A}[\mathcal{E}], \Gamma), \mathcal{F}(P_1'')) \in apply(\mathcal{F})(Moves(P_1')) \wedge \\
&\quad \mathcal{A} \cap \mathcal{L} = \emptyset|\!\} \uplus \\
&\quad \{\!|((\mathcal{A}[\mathcal{E}], \Gamma), \mathcal{F}(P_1') \underset{\mathcal{L}}{\bowtie} Q'') \mid ((\mathcal{A}[\mathcal{E}], \Gamma), Q'') \in apply(id)(Moves(Q')) \wedge \\
&\quad \mathcal{A} \cap \mathcal{L} = \emptyset|\!\} \uplus \\
&\quad \{\!|((\mathcal{A}[\mathcal{E}], \Gamma), \mathcal{F}(P_1'') \underset{\mathcal{L}}{\bowtie} Q'')) \mid ((\mathcal{A}_1[\mathcal{E}_1], \Gamma_1), \mathcal{F}(P_1'')) \in apply(\mathcal{F})(Moves(P_1')) \\
&\quad \wedge ((\mathcal{A}_2[\mathcal{E}_2], \Gamma_2), Q'') \in apply(id)(Moves(Q')) \wedge \mathcal{A}_1 \cap \mathcal{A}_2 \cap \mathcal{L} \neq \emptyset \wedge \\
&\quad \mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \wedge \mathcal{E} = \mathcal{E}_1 \uplus \mathcal{E}_2 \wedge \Gamma = \Gamma_1 \cup \Gamma_2|\!\} = \\
&apply(\mathcal{G})(Moves(P_1' \underset{\mathcal{L}}{\bowtie} Q')
\end{aligned}
$$

4. With the same procedure used for 3, we define a model process isomorphism $\mathcal{G}$ as: $\forall P_1' \underset{\mathcal{L}}{\bowtie} Q' \in ds(P_1 \underset{\mathcal{L}}{\bowtie} Q)$, with $P_1' \in ds(P_1)$ and $Q' \in ds(Q)$,

$$\mathcal{G}(P_1' \underset{\mathcal{L}}{\bowtie} Q') = \mathcal{F}(P_1') \underset{\mathcal{L}}{\bowtie} Q'$$

   $\mathcal{G}$ is a model process isomorphism because $\mathcal{F}$ is a model process isomorphism and it can be proved in analogy with point 3.

□

Proposition 6.23 $(\mathcal{T}, \Gamma)$*-isomorphism as a Congruence.* If $P_1, P_2 \in \mathbb{P}$ such that $P_1 \equiv_{\mathcal{T},\Gamma} P_2$, then

1. $\mathcal{A}[\mathcal{E}].P_1 \equiv_{\mathcal{T},\Gamma} \mathcal{A}[\mathcal{E}].P_2$, with P_1, P_2 agents

2. $P_1 + Q \equiv_{\mathcal{T},\Gamma} P_2 + Q$, with $P_1, P_2, Q \in \mathbb{P}_d$
3. $P_1 \underset{\mathcal{L}}{\rtimes} Q \equiv_{\mathcal{T},\Gamma} P_2 \underset{\mathcal{L}}{\rtimes} Q$, with $P_1, P_2, Q \in \mathbb{P}_m$
4. $P_1 \underset{\mathcal{L}}{\rtimes\!\!\rtimes} Q \equiv_{\mathcal{T},\Gamma} P_2 \underset{\mathcal{L}}{\rtimes\!\!\rtimes} Q$, with $P_1, P_2, Q \in \mathbb{P}_m$

Proof. Proof of each case:

1. Clearly $ds(\mathcal{A}[\mathcal{E}].P_1) = \{P_1\}$ and $ds(\mathcal{A}[\mathcal{E}].P_2) = \{P_2\}$. Thus we construct a bijective function $\mathcal{G} : ds(\mathcal{A}[\mathcal{E}].P_1) \to ds(\mathcal{A}[\mathcal{E}].P_2)$ as $\mathcal{G}(P_1) = P_2$, while we already have as an assumption that $P_1 \equiv_{\mathcal{T},\Gamma} P_2$ and so $P_1 \equiv_{\mathcal{T},\Gamma} \mathcal{G}(P_1)$. Finally, the only filtered composed action possible for both processes is $(\mathcal{A} \cap \mathcal{T})[\mathcal{E}]$. This implies that

$$FiltMoves(\mathcal{A}[\mathcal{E}].P_2)_{\mathcal{T}} = apply(\mathcal{G})(FiltMoves(\mathcal{A}[\mathcal{E}].P_1)_{\mathcal{T}})$$

2. We know $P_1, P_2 \in \mathbb{P}_d$ and there exists a bijective function $\mathcal{F} : ds(P_1) \to ds(P_2)$ s.t. $\forall A \in ds(P_1)$, $A \equiv_{\mathcal{T},\Gamma} \mathcal{F}(A)$ and

$$FiltMoves(P_2)_{\mathcal{T}} = apply(\mathcal{F})(FiltMoves(P_1)_{\mathcal{T}})$$

   Observe that $ds(P_1+Q) = ds(P_1) \cup ds(Q)$ and $ds(P_2+Q) = ds(P_2) \cup ds(Q)$. We construct $\mathcal{G} : ds(P_1 + Q) \to ds(P_2 + Q)$ as follows:

$$\mathcal{G}(A) = \begin{cases} \mathcal{F}(A) & \text{if } A \in ds(P_1) \\ A & \text{if } A \in ds(Q) \end{cases}$$

   this implies that $\forall A \in ds(P_1 + Q)$, $A \equiv_{\mathcal{T},\Gamma} \mathcal{G}(A)$ and

$$FiltMoves(P_2 + Q)_{\mathcal{T}} = FiltMoves(P_2)_{\mathcal{T}} \uplus FiltMoves(Q)_{\mathcal{T}} =$$

$$apply(\mathcal{F})(FiltMoves(P_1)_{\mathcal{T}}) \uplus apply(id)(FiltMoves(Q)_{\mathcal{T}}) =$$

$$apply(\mathcal{G})(FiltMoves(P_1 + Q)_{\mathcal{T}})$$

3. We know $P_1, P_2 \in \mathbb{P}_m$, and there exists an bijective function $\mathcal{F} : ds(P_1) \to$

$ds(P_2)$ s.t. $\mathcal{F}(P_1) = P_2$ and $\forall P' \in ds(P_1)$

$$FiltMoves(\mathcal{F}(P'))_{\mathcal{T},\Gamma} = apply(\mathcal{F})(FiltMoves(P')_{\mathcal{T},\Gamma})$$

We construct a $(\mathcal{T},\Gamma)$-isomorphism $\mathcal{G} : ds(P_1 \underset{\mathcal{L}}{\bowtie} Q) \rightarrow ds(P_2 \underset{\mathcal{L}}{\bowtie} Q)$ such that $\forall P' \underset{\mathcal{L}}{\bowtie} Q' \in ds(P_1 \underset{\mathcal{L}}{\bowtie} Q)$

$$FiltMoves(\mathcal{G}(P' \underset{\mathcal{L}}{\bowtie} Q'))_{\mathcal{T},\Gamma} = apply(\mathcal{G})(FiltMoves(P' \underset{\mathcal{L}}{\bowtie} Q')_{\mathcal{T},\Gamma})$$

We choose $\mathcal{G}(P' \underset{\mathcal{L}}{\bowtie} Q') = \mathcal{F}(P') \underset{\mathcal{L}}{\bowtie} Q'$. In fact:

$$\begin{aligned}
&FiltMoves(\mathcal{G}(P' \underset{\mathcal{L}}{\bowtie} Q'))_{\mathcal{T},\Gamma} = FiltMoves(\mathcal{F}(P') \underset{\mathcal{L}}{\bowtie} Q')_{\mathcal{T},\Gamma} = \\
&\quad \{\!|(a, R \underset{\mathcal{L}}{\bowtie} Q') \mid (a,R) \in FiltMoves(\mathcal{F}(P'))_{\mathcal{T},\Gamma}|\!\} \uplus \\
&\quad \{\!|(a, \mathcal{F}(P') \underset{\mathcal{L}}{\bowtie} Q'') \mid (a,Q'') \in FiltMoves(Q')_{\mathcal{T},\Gamma}|\!\} \uplus \\
&\quad \{\!|((\mathcal{A},\mathcal{E},r), R \underset{\mathcal{L}}{\bowtie} Q'')) \mid ((\mathcal{A}_1[\mathcal{E}_1],\Gamma_1),R) \in OpenMoves(\mathcal{F}(P')) \wedge \\
&\quad ((\mathcal{A}_2[\mathcal{E}_2],\Gamma_2),Q'') \in OpenMoves(Q')\mathcal{A}_1 \cap \mathcal{A}_2 \cap \mathcal{L} \neq \emptyset \wedge \\
&\quad (\mathcal{A}_1 \cup \mathcal{A}_2[\mathcal{E}_1 \uplus \mathcal{E}_2], \Gamma_1 \cup \Gamma_2) \in ClosedAct(\mathcal{F}(P') \underset{\mathcal{L}}{\bowtie} Q') \wedge \\
&\quad \{\!|(\mathcal{A},\mathcal{E},r)|\!\} = filterActivities(\mathcal{T})(rateActivities(\Gamma)(\\
&\quad \{\!|(\mathcal{A}_1 \cup \mathcal{A}_2[\mathcal{E}_1 \uplus \mathcal{E}_2], \Gamma_1 \cup \Gamma_2)|\!\}))|\!\} = \\
&apply(\mathcal{G})(FiltMoves(P' \underset{\mathcal{L}}{\bowtie} Q')_{\mathcal{T},\Gamma})
\end{aligned}$$

Notice that if an activity is filtered it will no longer synchronise via $\underset{\mathcal{L}}{\bowtie}$, because it is derived from a closed activity and because of condition 3 in Definition 5.12 (well formed process algebra with hooks models). Moreover, new filtered moves can be obtained only from the synchronisation of two open activities which form a single closed activity. The last $=$ is correct because

$$FiltMoves(\mathcal{F}(P'))_{\mathcal{T},\Gamma} = apply(\mathcal{F})(FiltMoves(P')_{\mathcal{T},\Gamma})$$

and because

$$OpenMoves(\mathcal{F}(P')) = apply(\mathcal{F})(OpenMoves(P'))$$

4. We know $P_1, P_2 \in \mathbb{P}_m$, and there exists an bijective function $\mathcal{F} : ds(P_1) \to ds(P_2)$ s.t. $\mathcal{F}(P_1) = P_2$ and $\forall P' \in ds(P_1)$

$$FiltMoves(\mathcal{F}(P'))_{\mathcal{T},\Gamma} = apply(\mathcal{F})(FiltMoves(P')_{\mathcal{T},\Gamma})$$

We construct a $(\mathcal{T},\Gamma)$-isomorphism $\mathcal{G} : ds(P_1 \underset{\mathcal{L}}{\bowtie} Q) \to ds(P_2 \underset{\mathcal{L}}{\bowtie} Q)$ such that $\forall P' \underset{\mathcal{L}}{\bowtie} Q' \in ds(P_1 \underset{\mathcal{L}}{\bowtie} Q)$

$$FiltMoves(\mathcal{G}(P' \underset{\mathcal{L}}{\bowtie} Q'))_{\mathcal{T},\Gamma} = apply(\mathcal{G})(FiltMoves(P' \underset{\mathcal{L}}{\bowtie} Q')_{\mathcal{T},\Gamma})$$

We choose $\mathcal{G}(P' \underset{\mathcal{L}}{\bowtie} Q') = \mathcal{F}(P') \underset{\mathcal{L}}{\bowtie} Q'$. In fact:

$$\begin{aligned}
&FiltMoves(\mathcal{G}(P' \underset{\mathcal{L}}{\bowtie} Q'))_{\mathcal{T},\Gamma} = FiltMoves(\mathcal{F}(P') \underset{\mathcal{L}}{\bowtie} Q')_{\mathcal{T},\Gamma} = \\
&\quad \{\!|((\mathcal{A},\mathcal{E},r), R \underset{\mathcal{L}}{\bowtie} Q') \mid ((\mathcal{A},\mathcal{E},r),R) \in FiltMoves(\mathcal{F}(P'))_{\mathcal{T},\Gamma} \wedge \neg(\exists \mathcal{B},\mathcal{E}', \\
&\quad \Gamma', Q'' \text{ s.t. } ((\mathcal{B}[\mathcal{E}'],\Gamma'),Q'') \in Moves(Q') \wedge \mathcal{B} \subseteq \mathcal{E} \cap \mathcal{L})\}\!\}\uplus \\
&\quad \{\!|((\mathcal{B},\mathcal{E}',r), \mathcal{F}(P') \underset{\mathcal{L}}{\bowtie} Q'') \mid ((\mathcal{B},\mathcal{E}',r),Q'') \in FiltMoves(Q')_{\mathcal{T},\Gamma} \wedge \neg(\exists \mathcal{A}, \\
&\quad \mathcal{E}, \Gamma', R \text{ s.t. } ((\mathcal{A}[\mathcal{E}],\Gamma'),R) \in Moves(\mathcal{F}(P')) \wedge \mathcal{A} \subseteq \mathcal{E}' \cap \mathcal{L})\}\!\}\uplus \\
&\quad \{\!|((\mathcal{A} \cup (\mathcal{B} \setminus \mathcal{T}), (\mathcal{E} \setminus \mathcal{B}) \uplus \mathcal{E}', r), R \underset{\mathcal{L}}{\bowtie} Q'') \mid ((\mathcal{A},\mathcal{E},r),R) \in \\
&\quad FiltMoves(\mathcal{F}(P'))_{\mathcal{T},\Gamma} \wedge ((\mathcal{B}[\mathcal{E}'],\Gamma_2),Q'') \in Moves(Q') \wedge \mathcal{B} \subseteq \mathcal{E} \cap \mathcal{L} \wedge \\
&\quad \neg(\exists \mathcal{B}', \mathcal{E}'', \Gamma_2', Q''' \text{ s.t. } ((\mathcal{B}'[\mathcal{E}''],\Gamma_2'),Q''') \in Moves(Q') \wedge \mathcal{B}' \subseteq \mathcal{E} \cap \mathcal{L} \wedge \\
&\quad |\mathcal{B}'| > |\mathcal{B}|)\}\!\}\uplus
\end{aligned}$$

$$\{\!|(((\mathcal{A} \setminus \mathcal{T}) \cup \mathcal{B}, (\mathcal{E}' \setminus \mathcal{A}) \uplus \mathcal{E}, r), R \underset{\mathcal{L}}{\bowtie} Q'') \mid ((\mathcal{A}, \mathcal{E}, \Gamma_1), R) \in$$

$$Moves(\mathcal{F}(P')) \wedge ((\mathcal{B}, \mathcal{E}', r), Q'') \in FiltMoves(Q')_{\mathcal{T},\Gamma} \wedge \mathcal{A} \subseteq \mathcal{E}' \cap \mathcal{L} \wedge$$

$$\neg(\exists \mathcal{A}', \mathcal{E}'', \Gamma_1', R' \text{ s.t. } ((\mathcal{A}'[\mathcal{E}''], \Gamma_1'), R') \in Moves(\mathcal{F}(P')) \wedge \mathcal{A}' \subseteq \mathcal{E}' \cap \mathcal{L} \wedge$$

$$|\mathcal{A}'| > |\mathcal{A}|)\!|\} =$$

$$apply(\mathcal{G})(FiltMoves(P' \underset{\mathcal{L}}{\bowtie} Q')_{\mathcal{T},\Gamma})$$

Notice that synchronisation via $\underset{\mathcal{L}}{\bowtie}$ only affects the action sets in the activity, leaving the rate unaltered. In particular, if $((\mathcal{A}, \mathcal{E}, r), P'')$ is a filtered move of P', this cannot synchronise with filtered moves of Q' via $\underset{\mathcal{L}}{\bowtie}$. This is because of condition 4 in Definition 5.12 and because $\mathcal{B} \subseteq \mathcal{L} \cap \mathcal{E}$ should hold for a move $((\mathcal{B}[\mathcal{E}'], \Gamma), Q'')$ of Q'. Because $\mathcal{E}$ cannot contain active actions (i.e. actions a such that $f_a \in \mathbb{F}$) then $\mathcal{B}$ does not as well and activity $(\mathcal{B}[\mathcal{E}'], \Gamma)$ is therefore open.

□

6.2.2 Markovian $(\mathcal{T}, \Gamma)$-bisimulation

In this section we define formally Markovian $(\mathcal{T}, \Gamma)$-bisimulation ($\simeq_{\mathcal{T},\Gamma}$) on sPAH processes. The definition is based on strong equivalence in PEPA (Hillston, 1996) and integrated equivalence in EMPA (Bernardo, 1996). Two sPAH processes are considered Markovian $(\mathcal{T}, \Gamma)$-bisimilar if it is possible to group the states of their filtered derivation graphs into equivalence classes in such a way that states belonging to the same equivalence class are characterised as follows:

- the sum of the rates of rated moves presenting the same action sets from a state in an equivalence class toward the states of another equivalence class is the same for all states in the same equivalence class;
- the set of open moves from a state in an equivalence class toward the states of another equivalence class is the same for all states in the same equivalence class.

In addition we prove that $\equiv_{\mathcal{T},\Gamma} \subset \simeq_{\mathcal{T},\Gamma}$ (Propositions 6.31 and 6.32), and we show that $\simeq_{\mathcal{T},\Gamma}$ is an equivalence relation (Proposition 6.28) and a congruence (Proposition 6.34).

Definition 6.24 *Functions $\mu_{\mathcal{T},\Gamma}$ and $\nu_{\mathcal{T},\Gamma}$.* Function $\mu_{\mathcal{T},\Gamma}$ returns the rate r at which a model process M can become M' with filtered transitions labelled with $(\mathcal{A}, \mathcal{E})$. Function $\nu_{\mathcal{T},\Gamma}$ returns instead the rate at which M can move to a set of model processes $\mathcal{C}$ with filtered transitions labelled with $(\mathcal{A}, \mathcal{E})$.

$$\mu_{\mathcal{T},\Gamma}(M, \mathcal{A}, \mathcal{E}, M') = \sum_{r_i \in I} r_i$$

where $I = \{\!|r \mid ((\mathcal{A}, \mathcal{E}, r), M') \in FiltMoves(M)_{\mathcal{T},\Gamma}|\!\}$. For each rate r, the same multiplicity of $((\mathcal{A}, \mathcal{E}, r), M')$ in $FiltMoves(M)_{\mathcal{T},\Gamma}$ is used.

$$\nu_{\mathcal{T},\Gamma}(M, \mathcal{A}, \mathcal{E}, \mathcal{C}) = \sum_{M' \in \mathcal{C}} \mu_{\mathcal{T},\Gamma}(M, \mathcal{A}, \mathcal{E}, M')$$

Definition 6.25 *Open Activities toward a set of model processes.* The multi-set of activities toward a set of model processes $\mathcal{C} \subseteq \mathbb{P}_m$ of a model process $M \in \mathbb{P}_m$ is defined as:

$$OpenAct(M, \mathcal{C}) = \{\!|(\mathcal{A}[\mathcal{E}], \Gamma) \mid ((\mathcal{A}[\mathcal{E}], \Gamma), M') \in OpenMoves(M) \wedge M' \in \mathcal{C}|\!\}$$

Definition 6.26 *Model process Markovian $(\mathcal{T}, \Gamma)$-bisimulation.* Given an action multi-set $\mathcal{T}$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, an equivalence relation over model processes $\mathcal{R} \subseteq \mathbb{P}_m \times \mathbb{P}_m$ is a model process Markovian $(\mathcal{T}, \Gamma)$-bisimulation iff whenever $(M_1, M_2) \in \mathcal{R}$ then $\forall$ action multi-sets $\mathcal{A}$ and $\mathcal{E}$ and $\forall \mathcal{C} \in \mathbb{P}_m/\mathcal{R}$

$$\nu_{\mathcal{T},\Gamma}(M_1, \mathcal{A}, \mathcal{E}, \mathcal{C}) = \nu_{\mathcal{T},\Gamma}(M_2, \mathcal{A}, \mathcal{E}, \mathcal{C})$$

and

$$OpenAct(M_1, \mathcal{C}) = OpenAct(M_2, \mathcal{C})$$

Definition 6.27 *Model process Markovian $(\mathcal{T}, \Gamma)$-bisimilarity.* Model process Markovian $(\mathcal{T}, \Gamma)$-bisimilarity, denoted $\simeq_{\mathcal{T},\Gamma}$, is the union of all model process Markovian $(\mathcal{T}, \Gamma)$-bisimulations, i.e.

$$\simeq_{\mathcal{T},\Gamma} = \bigcup \{\mathcal{R} \mid \mathcal{R} \text{ is a model process Markovian } (\mathcal{T}, \Gamma)\text{-bisim.}\}$$

Two model processes $M_1, M_2 \in \mathbb{P}_m$ are Markovian $(\mathcal{T}, \Gamma)$-bisimilar, denoted $M_1 \simeq_{\mathcal{T},\Gamma} M_2$, iff there is a model process $(\mathcal{T}, \Gamma)$-bisimulation $\mathcal{R}$ between them such that $(M_1, M_2) \in \mathcal{R}$.

If Γ is clear from the context, we write $M_1 \simeq_{\mathcal{T}} M_2$ instead of $M_1 \simeq_{\mathcal{T},\Gamma} M_2$ and we say M_1 and M_2 are Markovian $\mathcal{T}$-bisimilar.

Proposition 6.28 *Model process Markovian $(\mathcal{T}, \Gamma)$-bisimilarity is an equivalence relation.*

Proof. A model process Markovian $(\mathcal{T}, \Gamma)$-bisimilarity is an equivalence relation iff it is *symmetric*, *reflexive* and *transitive*. The first two properties are trivially true. We prove *transitivity*.

Prove that if $P \simeq_{\mathcal{T},\Gamma} Q$ and $Q \simeq_{\mathcal{T},\Gamma} R$ then $P \simeq_{\mathcal{T},\Gamma} R$.

$P \simeq_{\mathcal{T},\Gamma} Q$ implies there exists an equivalence relation over model processes $\mathcal{R}_1 \subseteq \mathbb{P}_m \times \mathbb{P}_m$ such that $(P, Q) \in \mathcal{R}_1$ and $\mathcal{R}_1$ is a model process Markovian $(\mathcal{T}, \Gamma)$-bisimulation. In the same way, $Q \simeq_{\mathcal{T},\Gamma} R$ implies there exists an equivalence relation over model processes $\mathcal{R}_2 \subseteq \mathbb{P}_m \times \mathbb{P}_m$ such that $(Q, R) \in \mathcal{R}_2$ and $\mathcal{R}_2$ is a model process Markovian $(\mathcal{T}, \Gamma)$-bisimulation. We prove that there exists an equivalence relation over model processes $\mathcal{R}_3 \subseteq \mathbb{P}_m \times \mathbb{P}_m$ such that $(P, R) \in \mathcal{R}_3$ and $\mathcal{R}_3$ is a model process Markovian $(\mathcal{T}, \Gamma)$-bisimulation.

We propose $\mathcal{R}_3$ such that $(P, R) \in \mathcal{R}_3 \Leftrightarrow (\exists Q \in \mathbb{P}_m$ s.t. $(P, Q) \in \mathcal{R}_1$ and $(Q, R) \in \mathcal{R}_2)$.

In fact, if $(P, R) \in \mathcal{R}_3$ then $\forall$ action multi-sets $\mathcal{A}$ and $\mathcal{E}$ and $\forall \mathcal{C}_3 \in \mathbb{P}_m/\mathcal{R}_3$

$$\nu_{\mathcal{T},\Gamma}(P, \mathcal{A}, \mathcal{E}, \mathcal{C}_3) = \nu_{\mathcal{T},\Gamma}(P, \mathcal{A}, \mathcal{E}, \mathcal{C}_1) = \nu_{\mathcal{T},\Gamma}(Q, \mathcal{A}, \mathcal{E}, \mathcal{C}_1)$$

$$= \nu_{\mathcal{T},\Gamma}(Q, \mathcal{A}, \mathcal{E}, \mathcal{C}_2) = \nu_{\mathcal{T},\Gamma}(R, \mathcal{A}, \mathcal{E}, \mathcal{C}_2) = \nu_{\mathcal{T},\Gamma}(R, \mathcal{A}, \mathcal{E}, \mathcal{C}_3)$$

and

$$OpenAct(P, \mathcal{C}_3) = OpenAct(P, \mathcal{C}_1) = OpenAct(Q, \mathcal{C}_1)$$

$$= OpenAct(Q, \mathcal{C}_2) = OpenAct(R, \mathcal{C}_2) = OpenAct(R, \mathcal{C}_3)$$

with $\mathcal{C}_1 \in \mathbb{P}_m/\mathcal{R}_1$, $\mathcal{C}_2 \in \mathbb{P}_m/\mathcal{R}_2$ and with $\mathcal{C}_1 \cap \mathcal{C}_2 \neq \emptyset$, $\mathcal{C}_1 \cap \mathcal{C}_3 \neq \emptyset$ and $\mathcal{C}_2 \cap \mathcal{C}_3 \neq \emptyset$.

□

Definition 6.29 *Filtered composed actions toward a set of model processes.* Given a multi-set of actions $\mathcal{T}$, the multi-set of composed actions toward a set of model processes $\mathcal{C} \subseteq \mathbb{P}_m$ of a definition process $D \in \mathbb{P}_d$ is defined as:

$$FiltCompAct(D, \mathcal{C})_{\mathcal{T}} = \{\!|(\mathcal{A}, \mathcal{E}) \mid ((\mathcal{A}, \mathcal{E}), D') \in FiltMoves(D)_{\mathcal{T}} \wedge D' \in \mathcal{C}|\!\}$$

Definition 6.30 *Markovian $(\mathcal{T}, \Gamma)$-bisimilar definition processes.* Given an action multi-set $\mathcal{T}$ and an environment $\Gamma \subseteq Names \times \mathbb{R}$, two definition processes $D_1, D_2 \in \mathbb{P}_d$ are definition process Markovian $(\mathcal{T}, \Gamma)$-bisimilar ($D_1 \simeq_{\mathcal{T},\Gamma} D_2$) iff $\forall \mathcal{C} \in \mathbb{P}_m / \simeq_{\mathcal{T},\Gamma}$

$$FiltCompAct(D_1, \mathcal{C})_{\mathcal{T}} = FiltCompAct(D_2, \mathcal{C})_{\mathcal{T}}$$

Proposition 6.31 Given $M_1, M_2 \in \mathbb{P}_m$,

$$\forall \Gamma \subseteq Names \times \mathbb{R},\ \forall \text{ multi-sets of actions } \mathcal{T},\ M_1 \equiv_{\mathcal{T},\Gamma} M_2 \implies M_1 \simeq_{\mathcal{T},\Gamma} M_2$$

Proof. We prove that $\equiv_{\mathcal{T},\Gamma}$ is a model process Markovian $(\mathcal{T}, \Gamma)$-bisimulation. Observe that $M_1 \equiv_{\mathcal{T},\Gamma} M_2$ iff $M_2 = \mathcal{F}(M_1)$ with $\mathcal{F}$ model process $(\mathcal{T}, \Gamma)$-isomorphism. By definition of model process $(\mathcal{T}, \Gamma)$-isomorphism we have that $\forall M_1' \in ds(M_1)$

$$FiltMoves(\mathcal{F}(M_1'))_{\mathcal{T},\Gamma} = apply(\mathcal{F})(FiltMoves(M_1')_{\mathcal{T},\Gamma})$$

and

$$OpenMoves(\mathcal{F}(M_1')) = apply(\mathcal{F})(OpenMoves(M_1'))$$

This implies that $\forall M'$ first derivative of M_1, $\forall M'' \in ds(M')$

$$FiltMoves(\mathcal{F}(M''))_{\mathcal{T},\Gamma} = apply(\mathcal{F})(FiltMoves(M'')_{\mathcal{T},\Gamma})$$

and

$$OpenMoves(\mathcal{F}(M'')) = apply(\mathcal{F})(OpenMoves(M''))$$

Which implies $M' \equiv_{\mathcal{T},\Gamma} \mathcal{F}(M')$. From this we derive that, $\forall$ multi-sets of actions

$\mathcal{A}$ and $\mathcal{E}$ and $\forall \mathcal{C} \in \mathbb{P}_m/\equiv_{\mathcal{T},\Gamma}$

$$\nu_{\mathcal{T},\Gamma}(M_1,\mathcal{A},\mathcal{E},\mathcal{C}) = \sum_{M'\in\mathcal{C}} \mu_{\mathcal{T},\Gamma}(M_1,\mathcal{A},\mathcal{H},M') = \sum_{M'\in\mathcal{C}} \mu_{\mathcal{T},\Gamma}(\mathcal{F}(M_1),\mathcal{A},\mathcal{H},\mathcal{F}(M'))$$
$$= \sum_{M'\in\mathcal{C}} \mu_{\mathcal{T},\Gamma}(M_2,\mathcal{A},\mathcal{H},M') = \nu_{\mathcal{T},\Gamma}(M_2,\mathcal{A},\mathcal{H},\mathcal{C})$$

and

$$OpenAct(M_1,\mathcal{C}) = OpenAct(M_2,\mathcal{C})$$

This is possible because M' and $\mathcal{F}(M')$ are $(\mathcal{T},\Gamma)$-isomorphic and belong to the same equivalence class $\mathcal{C} \in \mathbb{P}_m/\equiv_{\mathcal{T},\Gamma}$.

□

Proposition 6.32 Given $D_1, D_2 \in \mathbb{P}_d$,

$$\forall \Gamma \subseteq Names \times \mathbb{R},\ \forall \text{ multi sets of actions } \mathcal{T},\ D_1 \equiv_{\mathcal{T},\Gamma} D_2 \implies D_1 \simeq_{\mathcal{T},\Gamma} D_2$$

Proof. Given arbitrary $\Gamma \subseteq Names \times \mathbb{R}$ and $\mathcal{T} \subseteq Actions$, $D_1 \equiv_{\mathcal{T},\Gamma} D_2$ implies there exists a bijective function $\mathcal{F} : ds(D_1) \to ds(D_2)$ such that $\forall A \in ds(D_1)$, $A \equiv_{\mathcal{T},\Gamma} \mathcal{F}(A)$ and

$$FiltMoves(D_2)_{\mathcal{T}} = apply(\mathcal{F})(FiltMoves(D_1)_{\mathcal{T}})$$

We know from Proposition 6.31 that $A \equiv_{\mathcal{T},\Gamma} \mathcal{F}(A)$ implies $A \simeq_{\mathcal{T},\Gamma} \mathcal{F}(A)$. Thus, A and $\mathcal{F}(A)$ are in the same equivalence class $\mathcal{C} \in \mathbb{P}_m/\simeq_{\mathcal{T},\Gamma}$. This implies $\forall \mathcal{C} \in \mathbb{P}_m/\simeq_{\mathcal{T},\Gamma}$

$$FiltCompAct(D_1,\mathcal{C})_{\mathcal{T}} = FiltCompAct(D_2,\mathcal{C})_{\mathcal{T}}$$

□

Proposition 6.33 *Equational laws for Markovian* $(\mathcal{T},\Gamma)$*-bisimilar* sPAH *processes.* Because of Propositions 6.31 and 6.32, the equational laws for $(\mathcal{T},\Gamma)$-isomorphic sPAH processes hold as equational laws for Markovian $(\mathcal{T},\Gamma)$-bisimilar sPAH processes.

Proposition 6.34 *Markovian $(\mathfrak{T},\Gamma)$-bisimulation as a Congruence.* If $P_1, P_2 \in \mathbb{P}$ such that $P_1 \simeq_{\mathfrak{T},\Gamma} P_2$, then

1. $\mathcal{A}[\mathcal{E}].P_1 \simeq_{\mathfrak{T},\Gamma} \mathcal{A}[\mathcal{E}].P_2$, with P_1, P_2 agents
2. $P_1 + Q \simeq_{\mathfrak{T},\Gamma} P_2 + Q$, with $P_1, P_2, Q \in \mathbb{P}_d$
3. $P_1 \underset{\mathcal{L}}{\bowtie} Q \simeq_{\mathfrak{T},\Gamma} P_2 \underset{\mathcal{L}}{\bowtie} Q$, with $P_1, P_2, Q \in \mathbb{P}_m$
4. $P_1 \underset{\mathcal{L}}{\boxtimes} Q \simeq_{\mathfrak{T},\Gamma} P_2 \underset{\mathcal{L}}{\boxtimes} Q$, with $P_1, P_2, Q \in \mathbb{P}_m$

Proof. Proof of each case:

1. We know by assumption that $P_1 \simeq_{\mathfrak{T},\Gamma} P_2$. This implies $P_1, P_2 \in \mathcal{C}$, with $\mathcal{C} \in \mathbb{P}_m/\simeq_{\mathfrak{T},\Gamma}$. Now, $FiltMoves(\mathcal{A}[\mathcal{E}].P_1)_{\mathfrak{T}} = \{\!|(((\mathcal{A} \setminus \mathfrak{T}), \mathcal{E}), P_1)|\!\}$ and $FiltMoves(\mathcal{A}[\mathcal{E}].P_2)_{\mathfrak{T}} = \{\!|(((\mathcal{A} \setminus \mathfrak{T}), \mathcal{E}), P_2)|\!\}$. This implies $\forall \mathcal{C} \in \mathbb{P}_m/\simeq_{\mathfrak{T},\Gamma}$

$$FiltCompAct(\mathcal{A}[\mathcal{E}].P_1, \mathcal{C})_{\mathfrak{T}} = FiltCompAct(\mathcal{A}[\mathcal{E}].P_2, \mathcal{C})_{\mathfrak{T}}$$

2. $P_1 + Q$ and $P_2 + Q$ are definition processes. Because $P_1 \simeq_{\mathfrak{T},\Gamma} P_2$ we have that $\forall \mathcal{C} \in \mathbb{P}_m/\simeq_{\mathfrak{T},\Gamma}$

$$FiltCompAct(P_1 + Q, \mathcal{C})_{\mathfrak{T}} =$$

$$FiltCompAct(P_1, \mathcal{C})_{\mathfrak{T}} \uplus FiltCompAct(Q, \mathcal{C})_{\mathfrak{T}} =$$

$$FiltCompAct(P_2, \mathcal{C})_{\mathfrak{T}} \uplus FiltCompAct(Q, \mathcal{C})_{\mathfrak{T}} =$$

$$FiltCompAct(P_2 + Q, \mathcal{C})_{\mathfrak{T}}$$

3. We prove that the relation $\mathcal{R} = \{(P_1 \underset{\mathcal{L}}{\bowtie} Q, P_2 \underset{\mathcal{L}}{\bowtie} Q) \mid P_1 \simeq_{\mathfrak{T},\Gamma} P_2\}$ is a Markovian $(\mathfrak{T},\Gamma)$-bisimulation. We consider four cases:

   (a) $\forall r \in \mathbb{R}$, $(\mathcal{A}, \mathcal{E}, r) \notin FiltAct(P_1 \underset{\mathcal{L}}{\bowtie} Q)_{\mathfrak{T},\Gamma}$. This implies that $\forall r \in \mathbb{R}$, $(\mathcal{A}, \mathcal{E}, r) \notin FiltAct(P_2 \underset{\mathcal{L}}{\bowtie} Q)_{\mathfrak{T},\Gamma}$, because a filtered activity for $P \underset{\mathcal{L}}{\bowtie} Q$ is either a filtered activity of P, a filtered activity of Q or it is derived from a closed activity which is the synchronisation of an open activity from P and one from Q and $P_1 \simeq_{\mathfrak{T},\Gamma} P_2$ implies

- $\forall r \in \mathbb{R},\ (\mathcal{A}, \mathcal{E}, r) \notin FiltAct(P_1)_{\mathcal{T},\Gamma} \Leftrightarrow \forall r \in \mathbb{R},\ (\mathcal{A}, \mathcal{E}, r) \notin FiltAct(P_2)_{\mathcal{T},\Gamma}$;
- $\forall \mathcal{C} \in \mathbb{P}_m/\simeq_{\mathcal{T},\Gamma} OpenAct(P_1, \mathcal{C}) = OpenAct(P_2, \mathcal{C})$.

It follows that $\forall \mathcal{C} \in \mathbb{P}_m/\simeq_{\mathcal{T},\Gamma}$

$$\nu_{\mathcal{T},\Gamma}(P_1 \underset{\mathcal{L}}{\bowtie} Q, \mathcal{A}, \mathcal{E}, C) = \nu_{\mathcal{T},\Gamma}(P_2 \underset{\mathcal{L}}{\bowtie} Q, \mathcal{A}, \mathcal{E}, C) = 0$$

(b) $\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1' \underset{\mathcal{L}}{\bowtie} Q) \in FiltMoves(P_1 \underset{\mathcal{L}}{\bowtie} Q)_{\mathcal{T},\Gamma}$. Recall that $P_1 \simeq_{\mathcal{T},\Gamma} P_2$ implies

$$\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1') \in FiltMoves(P_1)_{\mathcal{T},\Gamma} \Leftrightarrow$$

$$\exists r' \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r'), P_2') \in FiltMoves(P_2)_{\mathcal{T},\Gamma}$$

with $P_1' \simeq_{\mathcal{T},\Gamma} P_2'$, because $\nu_{\mathcal{T},\Gamma}(P_1, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) = \nu_{\mathcal{T},\Gamma}(P_2, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}})$. It follows that

$$\nu_{\mathcal{T},\Gamma}(P_1 \underset{\mathcal{L}}{\bowtie} Q, \mathcal{A}, \mathcal{E}, [P_1' \underset{\mathcal{L}}{\bowtie} Q]_{\mathcal{R}}) = \nu_{\mathcal{T},\Gamma}(P_1, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) =$$

$$\nu_{\mathcal{T},\Gamma}(P_2, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) = \nu_{\mathcal{T},\Gamma}(P_2 \underset{\mathcal{L}}{\bowtie} Q, \mathcal{A}, \mathcal{E}, [P_2' \underset{\mathcal{L}}{\bowtie} Q]_{\mathcal{R}})$$

It is also important to recall that if an activity is filtered then it is derived from a closed activity, which in turn means that no further synchronisation is possible via $\underset{\mathcal{L}}{\bowtie}$. This is because of Condition 3 in Definition 5.12.

(c) $\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1 \underset{\mathcal{L}}{\bowtie} Q') \in FiltMoves(P_1 \underset{\mathcal{L}}{\bowtie} Q)_{\mathcal{T},\Gamma}$. It follows that

$$\nu_{\mathcal{T},\Gamma}(P_1 \underset{\mathcal{L}}{\bowtie} Q, \mathcal{A}, \mathcal{E}, [P_1 \underset{\mathcal{L}}{\bowtie} Q']_{\mathcal{R}}) = \nu_{\mathcal{T},\Gamma}(Q, \mathcal{A}, \mathcal{E}, [Q']_{\simeq_{\mathcal{T},\Gamma}}) =$$

$$= \nu_{\mathcal{T},\Gamma}(P_2 \underset{\mathcal{L}}{\bowtie} Q, \mathcal{A}, \mathcal{E}, [P_1 \underset{\mathcal{L}}{\bowtie} Q']_{\mathcal{R}})$$

Because $P_1 \simeq_{\mathcal{T},\Gamma} P_2$ and so $P_2 \underset{\mathcal{L}}{\bowtie} Q' \in [P_1 \underset{\mathcal{L}}{\bowtie} Q']_{\mathcal{R}}$.

(d) $\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1' \underset{\mathcal{L}}{\bowtie} Q') \in FiltMoves(P_1 \underset{\mathcal{L}}{\bowtie} Q)_{\mathcal{T},\Gamma}$. The filtered move $((\mathcal{A}, \mathcal{E}, r), P_1' \underset{\mathcal{L}}{\bowtie} Q')$ must be the result of a synchronisation between an open move from P_1 and an open move from Q. In particular, it must be that $\exists (\mathcal{A}_1[\mathcal{E}_1], \Gamma_1) \in OpenAct(P_1, [P_1']_{\simeq_{\mathcal{T},\Gamma}})$ and

$\exists(\mathcal{A}_2[\mathcal{E}_2], \Gamma_2) \in OpenAct(Q, [Q']_{\simeq_{\mathcal{T},\Gamma}})$ such that:

- $\mathcal{B} = \mathcal{A}_1 \cup \mathcal{A}_2$, $\mathcal{E} = \mathcal{E}_1 \uplus \mathcal{E}_2$ and $\Gamma' = \Gamma_1 \cup \Gamma_2$;
- $(\mathcal{B}[\mathcal{E}], \Gamma') \in ClosedAct(P_1 \underset{\mathcal{L}}{\bowtie} Q)$;
- $\{a\} = activeActions(\mathcal{B})_{\mathbb{F}}$;
- $\{\!|(\mathcal{A}, \mathcal{E}, k)|\!\} = filterActivities(\mathcal{T})(rateActivities(\Gamma)(\{\!|(\mathcal{B}[\mathcal{E}], \Gamma')|\!\}))$;
- $r = k/\pi(ClosedAct(P_1 \underset{\mathcal{L}}{\bowtie} Q), (a, \Gamma'))$.

Moreover, from $P_1 \simeq_{\mathcal{T},\Gamma} P_2$ we have that

$$((\mathcal{A}_1[\mathcal{E}_1], \Gamma_1), P_1') \in OpenMoves(P_1) \Leftrightarrow$$

$$((\mathcal{A}_1[\mathcal{E}_1], \Gamma_1), P_2') \in OpenMoves(P_2)$$

with $P_1' \simeq_{\mathcal{T},\Gamma} P_2'$, because $OpenAct(P_1, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) = OpenAct(P_2, [P_1']_{\simeq_{\mathcal{T},\Gamma}})$. It follows that

$$\nu_{\mathcal{T},\Gamma}(P_1 \underset{\mathcal{L}}{\bowtie} Q, \mathcal{A}, \mathcal{E}, [P_1' \underset{\mathcal{L}}{\bowtie} Q']_{\mathcal{R}}) = \nu_{\mathcal{T},\Gamma}(P_2 \underset{\mathcal{L}}{\bowtie} Q, \mathcal{A}, \mathcal{E}, [P_1' \underset{\mathcal{L}}{\bowtie} Q']_{\mathcal{R}})$$

where $P_2' \underset{\mathcal{L}}{\bowtie} Q' \in [P_1' \underset{\mathcal{L}}{\bowtie} Q']_{\mathcal{R}}$.

4. We prove that the relation $\mathcal{R} = \{(P_1 \underset{\mathcal{L}}{\boxtimes} Q, P_2 \underset{\mathcal{L}}{\bowtie} Q) \mid P_1 \simeq_{\mathcal{T},\Gamma} P_2\}$ is a Markovian $(\mathcal{T}, \Gamma)$-bisimulation. We consider four cases:

   (a) $\forall r \in \mathbb{R}$, $(\mathcal{A}, \mathcal{E}, r) \notin FiltAct(P_1 \underset{\mathcal{L}}{\boxtimes} Q)_{\mathcal{T},\Gamma}$. This implies that $\forall r \in \mathbb{R}$, $(\mathcal{A}, \mathcal{E}, r) \notin FiltAct(P_2 \underset{\mathcal{L}}{\boxtimes} Q)_{\mathcal{T},\Gamma}$, because a filtered activity for $P \underset{\mathcal{L}}{\boxtimes} Q$ is either a filtered activity of P, a filtered activity of Q or it is derived from a synchronisation of a filtered activity from P and an open activity from Q or a filtered activity from Q and an open activity from P and $P_1 \simeq_{\mathcal{T},\Gamma} P_2$ implies

   - $\forall r \in \mathbb{R}$, $(\mathcal{A}, \mathcal{E}, r) \notin FiltAct(P_1)_{\mathcal{T},\Gamma} \Leftrightarrow \forall r \in \mathbb{R}$, $(\mathcal{A}, \mathcal{E}, r) \notin FiltAct(P_2)_{\mathcal{T},\Gamma}$;
   - $\forall \mathcal{C} \in \mathbb{P}_m/\simeq_{\mathcal{T},\Gamma} OpenAct(P_1, \mathcal{C}) = OpenAct(P_2, \mathcal{C})$.

   It follows that $\forall \mathcal{C} \in \mathbb{P}_m/\simeq_{\mathcal{T},\Gamma}$

$$\nu_{\mathcal{T},\Gamma}(P_1 \underset{\mathcal{L}}{\boxtimes} Q, \mathcal{A}, \mathcal{E}, C) = \nu_{\mathcal{T},\Gamma}(P_2 \underset{\mathcal{L}}{\boxtimes} Q, \mathcal{A}, \mathcal{E}, C) = 0$$

(b) $\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1' \underset{\mathcal{L}}{\boxtimes} Q) \in FiltMoves(P_1 \underset{\mathcal{L}}{\boxtimes} Q)_{\mathcal{T},\Gamma}$. This implies that $((\mathcal{A}, \mathcal{E}, r), P_1') \in FiltMoves(P_1)_{\mathcal{T},\Gamma}$ and that $\neg(\exists \mathcal{B}, \mathcal{F}, \Gamma', Q'$ s.t. $((\mathcal{B}[\mathcal{F}], \Gamma'), Q') \in Moves(Q) \wedge \mathcal{B} \subseteq \mathcal{E} \cap \mathcal{L})$. Moreover,

$$\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1') \in FiltMoves(P_1)_{\mathcal{T},\Gamma} \Leftrightarrow$$

$$\exists r' \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r'), P_2') \in FiltMoves(P_2)_{\mathcal{T},\Gamma}$$

with $P_1' \simeq_{\mathcal{T},\Gamma} P_2'$, because $\nu_{\mathcal{T},\Gamma}(P_1, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) = \nu_{\mathcal{T},\Gamma}(P_2, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}})$. It follows that

$$\nu_{\mathcal{T},\Gamma}(P_1 \underset{\mathcal{L}}{\boxtimes} Q, \mathcal{A}, \mathcal{E}, [P_1' \underset{\mathcal{L}}{\boxtimes} Q]_{\mathcal{R}}) = \nu_{\mathcal{T},\Gamma}(P_1, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) =$$

$$\nu_{\mathcal{T},\Gamma}(P_2, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) = \nu_{\mathcal{T},\Gamma}(P_2 \underset{\mathcal{L}}{\boxtimes} Q, \mathcal{A}, \mathcal{E}, [P_1' \underset{\mathcal{L}}{\boxtimes} Q]_{\mathcal{R}})$$

(c) $\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1 \underset{\mathcal{L}}{\boxtimes} Q') \in FiltMoves(P_1 \underset{\mathcal{L}}{\boxtimes} Q)_{\mathcal{T},\Gamma}$. This implies that $((\mathcal{A}, \mathcal{E}, r), Q') \in FiltMoves(Q)_{\mathcal{T},\Gamma}$ and that $\neg(\exists \mathcal{B}, \mathcal{F}, \Gamma', P_1'$ s.t. $((\mathcal{B}[\mathcal{F}], \Gamma'), P_1') \in Moves(P_1) \wedge \mathcal{B} \subseteq \mathcal{E} \cap \mathcal{L})$.
Because $\forall \mathcal{C} \in \mathbb{P}_m/\simeq_{\mathcal{T},\Gamma},\ OpenAct(P_1, \mathcal{C}) = OpenAct(P_2, \mathcal{C})$ then

$$\nu_{\mathcal{T},\Gamma}(P_1 \underset{\mathcal{L}}{\boxtimes} Q, \mathcal{A}, \mathcal{E}, [P_1 \underset{\mathcal{L}}{\boxtimes} Q']_{\mathcal{R}}) = \nu_{\mathcal{T},\Gamma}(Q, \mathcal{A}, \mathcal{E}, [Q']_{\simeq_{\mathcal{T},\Gamma}}) =$$

$$\nu_{\mathcal{T},\Gamma}(P_2 \underset{\mathcal{L}}{\boxtimes} Q, \mathcal{A}, \mathcal{E}, [P_1 \underset{\mathcal{L}}{\boxtimes} Q']_{\mathcal{R}})$$

(d) $\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1' \underset{\mathcal{L}}{\boxtimes} Q') \in FiltMoves(P_1 \underset{\mathcal{L}}{\boxtimes} Q)_{\mathcal{T},\Gamma}$. This implies that filtered move $((\mathcal{A}, \mathcal{E}, r), P_1' \underset{\mathcal{L}}{\boxtimes} Q')$ is the result of the synchronisation of a filtered move of P_1 and an open move of Q or the synchronisation of a filtered move from Q and an open move from P_1.
In this case we have

$$\nu_{\mathcal{T},\Gamma}(P_1 \underset{\mathcal{L}}{\boxtimes} Q, \mathcal{A}, \mathcal{E}, [P_1' \underset{\mathcal{L}}{\boxtimes} Q']_{\mathcal{R}}) =$$

$$\sum_{i \in I} \nu_{\mathcal{T},\Gamma}(P_1, \mathcal{A}_i, \mathcal{E}_i, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) + \sum_{j \in J} \nu_{\mathcal{T},\Gamma}(Q, \mathcal{A}_j, \mathcal{E}_J, [Q']_{\simeq_{\mathcal{T},\Gamma}}) =$$

$$\sum_{i \in I} \nu_{\mathcal{T},\Gamma}(P_2, \mathcal{A}_i, \mathcal{E}_i, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) + \sum_{j \in J} \nu_{\mathcal{T},\Gamma}(Q, \mathcal{A}_j, \mathcal{E}_J, [Q']_{\simeq_{\mathcal{T},\Gamma}}) =$$

$$\nu_{\mathcal{T},\Gamma}(P_2 \underset{\mathcal{L}}{\boxtimes} Q, \mathcal{A}, \mathcal{E}, [P_1' \underset{\mathcal{L}}{\boxtimes} Q']_{\mathcal{R}})$$

Once again because

$$\exists r \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r), P_1') \in FiltMoves(P_1)_{\mathcal{T},\Gamma} \Leftrightarrow$$

$$\exists r' \in \mathbb{R},\ ((\mathcal{A}, \mathcal{E}, r'), P_2') \in FiltMoves(P_2)_{\mathcal{T},\Gamma}$$

with $P_1' \simeq_{\mathcal{T},\Gamma} P_2'$, because $\nu_{\mathcal{T},\Gamma}(P_1, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}}) = \nu_{\mathcal{T},\Gamma}(P_2, \mathcal{A}, \mathcal{E}, [P_1']_{\simeq_{\mathcal{T},\Gamma}})$. Moreover

- $i \in I$ if and only if $\exists \mathcal{B}, \mathcal{F}, \Gamma', Q'$ s.t. $((\mathcal{B}[\mathcal{F}], \Gamma'), Q') \in Moves(Q)$ and $\mathcal{B} \subseteq \mathcal{E}_i \cap \mathcal{L}$ and $\mathcal{A} = \mathcal{A}_i \cup (\mathcal{B} \cap \mathcal{T})$ and $\mathcal{E} = (\mathcal{E}_i \setminus \mathcal{B}) \uplus \mathcal{F}$ and $\neg(\exists \mathcal{B}', \mathcal{F}', \Gamma'', Q''$ s.t. $((\mathcal{B}'[\mathcal{F}'], \Gamma''), Q'') \in Moves(Q)$ and $\mathcal{B}' \subseteq \mathcal{E}_i \cap \mathcal{L}$ and $|\mathcal{B}'| > |\mathcal{B}|)$;
- $j \in J$ if and only if $\exists \mathcal{B}, \mathcal{F}, \Gamma', P_1'$ s.t. $((\mathcal{B}[\mathcal{F}], \Gamma'), P_1') \in Moves(P)$ and $\mathcal{B} \subseteq \mathcal{E}_j \cap \mathcal{L}$ and $\mathcal{A} = \mathcal{A}_j \cup (\mathcal{B} \cap \mathcal{T})$ and $\mathcal{E} = (\mathcal{E}_j \setminus \mathcal{B}) \uplus \mathcal{F}$ and $\neg(\exists \mathcal{B}', \mathcal{F}', \Gamma'', Q''$ s.t. $((\mathcal{B}'[\mathcal{F}'], \Gamma''), P_1'') \in Moves(P_1)$ and $\mathcal{B}' \subseteq \mathcal{E}_j \cap \mathcal{L}$ and $|\mathcal{B}'| > |\mathcal{B}|)$.

We also use the fact that $\forall \mathcal{C} \in \mathbb{P}_m / \simeq_{\mathcal{T},\Gamma},\ OpenAct(P_1, \mathcal{C}) = OpenAct(P_2, \mathcal{C})$.

Since we proved all four cases, the result holds.

□

6.2.3 Practical Use of the Relations

The three relations defined in this chapter represent the foundation of the theory of relations on sPAH. They are the strongest relations one can define, considering in turn the structure of the model (isomorphism), filtering of activities ($\mathcal{T}$-isomorphism) and timing and probability of actions (Markovian $\mathcal{T}$-bisimulation). In a case study in Section 7.2, we will illustrate how the fact that Markovian $\mathcal{T}$-bisimulation is a congruence can be exploited to extend the equality of parts of two model to the entire models.

As it can be expected from such strong relations, they may be too strong for many biological applications, but other relations can be defined using these as a starting point. In particular, in biology one is often interested in knowing whether two systems are *almost* rather than *exactly* the same, and possibly *to which extent* they are similar. We will discuss more about this in the last chapter, in the future work section.

6.3 Summary

In this chapter we discussed the idea of relating quantitative multi-scale models of biological systems at a specified scale using process algebra with hooks. After introducing *filtering* as a procedure that can be used to focus on a specified scale, we defined three relations on process algebra with hooks models: isomorphism ($\equiv$), $\mathcal{T}$-isomorphism ($\equiv_{\mathcal{T}}$) and Markovian $\mathcal{T}$-bisimulation ($\simeq_{\mathcal{T}}$). In turn, isomorphism relates processes that produce isomorphic derivation graphs, $\mathcal{T}$-isomorphism relates processes that produce derivation graphs that are isomorphic after rating and filtering and Markovian $\mathcal{T}$-bisimulation relates processes that perform the same filtered actions with the same probability at the same time. Fundamental properties of the relations, such as equational laws and congruence are also provided. Although these relations are probably too strong to be of practical use in the biological setting, they nevertheless can be considered the fundamental relations, the starting point from which other relations can be defined.

Chapter 7

Case Study

In this chapter we illustrate how process algebra with hooks can be employed to model complex multi-scale scenarios. Before presenting the models we introduce a parametric version of stochastic process algebra with hooks (Section 7.1); this is a minor extension, which does not affect the theory of earlier chapters, but makes model descriptions more compact. The two scenarios we model in this chapter are: a typical problem of pattern formation (Section 7.2), and a multi-scale model of tissue growth (Section 7.3). We perform analysis using an interpreter and simulator developed for parametric stochastic process algebra with hooks.

7.1 Parametric Stochastic Process Algebra with Hooks

In this section we introduce the syntax of a parametric version of sPAH. We augment sPAH defined in Section 5.2 with parametrised processes and actions, in analogy with our definition of pSPA in Section 3.3, and we obtain a parametric stochastic process algebra with hooks (psPAH). The syntax of psPAH is as follows:

$$
\begin{aligned}
D ::= \;& nil \mid \mathcal{L}'[\mathcal{L}''].A(exp, \ldots, exp) \mid D + D \mid \text{if } bexp \text{ then } D \text{ else } D \\
M ::= \;& A(k, \ldots, k) \mid M \underset{\mathcal{L}}{\bowtie} M \mid M \underset{\mathcal{L}}{\boxtimes} M \\
exp ::= \;& k \mid i \mid exp + exp \mid exp - exp \mid exp/exp \mid exp * exp \\
bexp ::= \;& exp = exp \mid exp < exp \mid bexp \wedge bexp \mid \\
& bexp \vee bexp \mid \neg bexp \mid true \mid false
\end{aligned}
$$

The main differences between this and the non parametric version are:

- actions have the form $a(exp, \ldots, exp)$, where $exp, \ldots, exp$ is a list of expressions;
- $k \in \mathbb{R}$ and i is a parameter name, i.e. $i \in Names$;
- a definition process can also be an if-then-else construct: if $bexp$ then D else D;
- agent definitions have now the form $A(i_1, \ldots, i_n) \triangleq D$, where $i_1, \ldots, i_n$ is a list of parameter names;
- the evaluation of the expressions is performed when inference rule **Agent** is applied;
- the definitions of functional rates and the variables associated with agents are also parametric.

The semantics is given in Figure 7.1. Given an environment Γ, the evaluation of an expression exp into a real number k is denoted by $\Gamma \vdash exp \to k$, the evaluation of a boolean expression $bexp$ into $b \in \{true, false\}$ is denoted by $\Gamma \vdash bexp \to b$, while the evaluation of the list of expressions of all the actions in a set $\mathcal{A}$ is denoted by $\Gamma \vdash \mathcal{A} \to \mathcal{A}'$, where $\mathcal{A}'$ contains only actions with evaluated expressions.

An interpreter for psPAH has been implemented in the functional programming language OCaml. The interpreter reads as input the description of a psPAH model along with a model time threshold for the simulations. Simulations are performed on a model, producing traces of states and time delays using the sampling method for rated LTSs (Section 2.3.2).

Simulations have been performed on a laptop computer with Ubuntu Linux, two Intel Core 2 Duo 2.20 GHz CPUs and 2 GB of RAM.

In the following examples the number of simulations performed is chosen to be reasonable to obtain an accurate analysis of the models in the reasonable time of one or two working days.

Prefix

$$\frac{}{\mathcal{A}[\mathcal{E}].A(exp_1,\ldots,exp_n) \xrightarrow{\mathcal{A}[\mathcal{E}],true} A(exp_1,\ldots,exp_n)}$$

Choice Left

$$\frac{D_1 \xrightarrow{\mathcal{A}[\mathcal{E}],b} A(exp_1,\ldots,exp_n)}{D_1 + D_2 \xrightarrow{\mathcal{A}[\mathcal{E}],b} A(exp_1,\ldots,exp_n)}$$

Choice Right

$$\frac{D_2 \xrightarrow{\mathcal{A}[\mathcal{E}],b} A(exp_1,\ldots,exp_n)}{D_1 + D_2 \xrightarrow{\mathcal{A}[\mathcal{E}],b} A(exp_1,\ldots,exp_n)}$$

If Then Else True

$$\frac{D_1 \xrightarrow{\mathcal{A}[\mathcal{E}],b} A(exp_1,\ldots,exp_n)}{\text{if } bexp \text{ then } D_1 \text{ else } D_2 \xrightarrow{\mathcal{A}[\mathcal{E}],b\wedge bexp} A(exp_1,\ldots,exp_n)}$$

If Then Else False

$$\frac{D_2 \xrightarrow{\mathcal{A}[\mathcal{E}],b} A(exp_1,\ldots,exp_n)}{\text{if } bexp \text{ then } D_1 \text{ else } D_2 \xrightarrow{\mathcal{A}[\mathcal{E}],b\wedge\neg bexp} A(exp_1,\ldots,exp_n)}$$

Agent

$$\frac{D \xrightarrow{\mathcal{A}[\mathcal{E}],b} A'(exp_1,\ldots,exp_n)}{A(k_1,\ldots,k_n) \xrightarrow{(\mathcal{A}'[\mathcal{E}'],\Gamma')} A'(k'_1,\ldots,k'_n)}\ *$$

* if $A(i_1,\ldots,i_n) \triangleq D \wedge \Gamma = \{(i_1,k_1),\ldots,(i_n,k_n)\} \wedge \Gamma \vdash b \to true$
$\Gamma \vdash exp_1 \to k'_1 \wedge \cdots \wedge \Gamma \vdash exp_n \to k'_n \wedge \Gamma \vdash \mathcal{A} \to \mathcal{A}' \wedge \Gamma \vdash \mathcal{E} \to \mathcal{E}'$
$\wedge\ \Gamma' = \{(Var(A(k_1,\ldots,k_n)), Val(A(k_1,\ldots,k_n)))\}$

Figure 7.1: Semantics of parametric stochastic process algebra with hooks. Other inference rules are as in Figure 5.5.

7.2 Multi-Scale Model of Pattern Formation

In this section we give a psPAH specification of the French Flag Model (Wolpert, 1968), an example of how a group of identical cells can be differentiated into subgroups with different specialisations using positional information. For the biological background of this section see Section 2.1.4.

A morphogen **M** diffuses from a source into a tissue. In the long run, the region close to the source presents a high concentration of **M**, while the further a region is from the source, the lower the concentration of **M** is in that region. The concentration of **M** in a region indicates the positional information, i.e. the

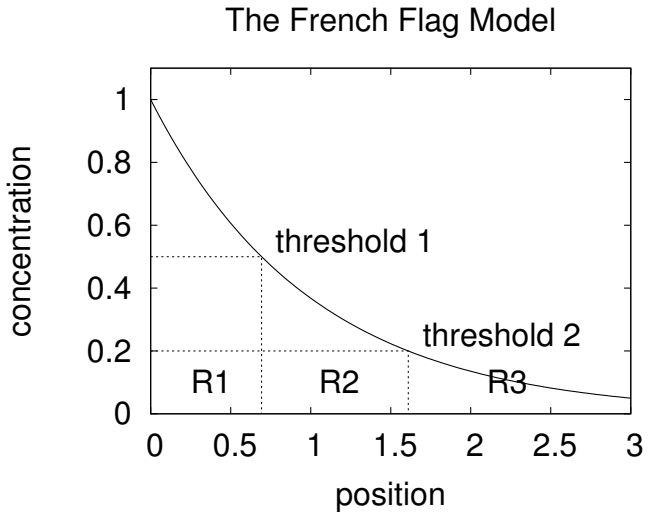


Figure 7.2: The French Flag Model implemented with partial differential equations. In the picture, two concentration thresholds divide the space into three regions.

position with respect to the source. Different specialisations are assigned to a region depending on the concentration of $\mathbf{M}$ in that region. Of great importance are concentration thresholds that delimit the concentration ranges associated with different specialisations.

One of the simplest models of this scenario is the following partial differential equation (PDE) (Alon, 2006):

$$\frac{\partial \mathbf{M}(t,x)}{\partial t} = D\frac{\partial^2 \mathbf{M}(t,x)}{\partial x^2} - \alpha \mathbf{M}(t,x)$$

The above equation models the concentration of $\mathbf{M}$ in time and space in one-dimensional coordinates. The element $D\frac{\partial^2 \mathbf{M}(t,x)}{\partial x^2}$ is the diffusion of $\mathbf{M}$, while $\alpha \mathbf{M}(t,x)$ is its degradation. Constant D is the diffusion constant and α is the degradation constant; we assume these two constants to be equal to 1. Boundary conditions are:

1. $\mathbf{M}(t,0) = 1$ with $0 \leq t < \infty$, i.e. a constant source of $\mathbf{M}$ at position 0;
2. $\mathbf{M}(t,\infty) = 0$ with $0 \leq t < \infty$, i.e. concentration of $\mathbf{M}$ is lost in the surroundings.

Steady state solution of the PDE model is shown in Figure 7.2. In the figure, the steady state solution is shown (continuous line). Two concentration thresholds

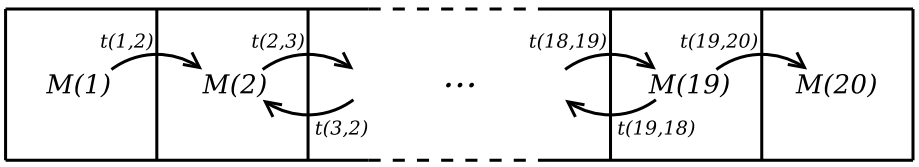


Figure 7.3: Discretisation of the space of the French Flag Model into 20 regions. The variable $M(i)$ indicates the concentration of **M** at region 1.

(at 0.2 and 0.5, dotted lines) divide the area in three regions (R1, R2 and R3). Each of these regions are characterised by a different specialisation.

We propose now a representation of the PDE model in psPAH. Because the PDE model is continuous in the concentration of **M** and in space, while psPAH uses a discrete representation of these quantities, we provide a discretisation. We assume 20 levels for the concentration of **M** (parameter $maxLevels = 20$), with maximum concentration equal to 1 and concentration of each level equal to $h = 1/20 = 0.05$, and we assume 20 regions of space (parameter $regions = 20$), with total length equal to 3 and length of each region equal to $deltaX = 3/20 = 0,15$. The spatial discretisation is illustrated in Figure 7.3.

In order to represent the two boundary conditions we have:

- the left most region presents a constant concentration level of 20, which guarantees that concentration flows continuously from the left;
- the right most region presents a constant concentration level equal to 0, which implies that this region absorbs concentration levels.

We define agent $M(i, w)$ (Figure 7.4) to indicate that morphogen **M** in region i presents concentration level w. Actions $t(i, j)$ represent transport of **M** from region i to region j, while actions $deg(i)$ represent degradation of **M** in region i.

We model specialisation of regions explicitly as follows. Two thresholds are considered: one between 4 and 5 concentration levels and the other between 10 and 11. Whenever a concentration threshold is crossed in a region, the specialisation of the region changes. The presence of two thresholds implies that three specialisations are possible, corresponding to high, medium and low concentration ranges of **M**. In addition, we consider that regions of tissue can commit permanently to a specialisation, if the concentration of **M** in those regions stays at a certain concentration range long enough. A commitment means that the cells in the committed region have memorised their positional information and further

```
M(i,w) ≜
if i == 1 then    //first region, the source of M
  t(1,2).M(1,w)
else
  if i == regions then    //last region, absorbing
    t(regions − 1, regions).M(regions, w)
  else    //any other region
    if w > 0 then    //degradation of M
      if w == (thr1 + 1) ∨ w == (thr2 + 1) then
        deg(i)[y(i)].M(i, w − 1)
      else
        deg(i).M(i, w − 1)
    else
      nil
    +    //transport of M to next region
    if i < regions ∧ w > 0 then
      if w == (thr1 + 1) ∨ w == (thr2 + 1) then
        t(i, i + 1)[y(i)].M(i, w − 1)
      else
        t(i, i + 1).M(i, w − 1)
    else
      nil
    +    //transport of M from next region
    if i < (regions − 1) ∧ w < maxLevels then
      if w == (thr1) ∨ w == (thr2) then
        t(i + 1, i)[x(i)].M(i, w + 1)
      else
        t(i + 1, i).M(i, w + 1)
    else
      nil
    +    //transport of M to previous region
    if i > 2 ∧ w > 0 then
      if w == (thr1 + 1) ∨ w == (thr2 + 1) then
        t(i, i − 1)[y(i)].M(i, w − 1)
      else
        t(i, i − 1).M(i, w − 1)
    else
      nil
    +    //transport of M from previous region
    if i > 1 ∧ w < maxLevels then
      if w == (thr1) ∨ w == (thr2) then
        t(i − 1, i)[x(i)].M(i, w + 1)
      else
        t(i − 1, i).M(i, w + 1)
    else
      nil
```

```
T(i,z,w) ≜
if w < 2 then
  x(i).T(i, 0, w + 1)
else
  nil
+
if w > 0 then
  (y(i).T(i, 0, w − 1)+
  if z < 1 then
    mem(i).T(i, z + 1, w)
  else
    if w == 2 then
      mem(i).TA(i)
    else
      mem(i).TB(i)
else
  nil


TA(i) ≜ nil


TB(i) ≜ nil
```

Figure 7.4: Agent definitions of processes $M(i,w)$, $T(i,z,w)$, $TA(i)$ and $TB(i)$, from the multi-scale model of pattern formation.

changes to the concentration of **M** will not affect the chosen specialisation. We use agent $T(i, z, w)$ (Figure 7.4) to represent tissue region i with specialisation w, while parameter z is part of the implementation of the commitment procedure. Parameter w is equal to 2 when the concentration of **M** is high, 1 when it is medium and 0 when it is low. Inter-scale actions $x(i)$ and $y(i)$ are used to synchronise processes $M(i, w)$ and $T(i, z, w)$. To obtain a commitment of a region we define a two-step memorisation:

1. when w is 1 or 2, $T(i, 0, w)$ can perform action $mem(i)$ and become $T(i, 1, w)$;
2. if $T(i, 1, w)$ performs $mem(i)$ then it becomes $TA(i)$ or $TB(i)$, depending on whether w is 2 or 1 respectively. Agents $TA(i)$ and $TB(i)$ are deadlock processes that represent the commitment of tissue region i to specialisations A and B, respectively.

In addition, if $T(i, 1, w)$ changes specialisation from w to w' then the process becomes $T(i, 0, w')$. This implies that the attempt at memorising specialisation w is forgotten and a new attempt at memorising specialisation w' can begin. The agent definitions of the French Flag Model are shown in Figure 7.4.

The initial state of the model is given by the following model component:

$$(M(1,20) \underset{\{|x(1),y(1)|\}}{\bowtie} TA(1)) \underset{\{|t(1,2)|\}}{\bowtie} (M(2,0) \underset{\{|x(2),y(2)|\}}{\bowtie} T(2,0,0)) \cdots \\ \cdots \underset{\{|t(19,20)|\}}{\bowtie} (M(20,0) \underset{\{|x(20),y(20)|\}}{\bowtie} T(20,0,0))$$

Functional rates for actions $t(i, j)$ and $deg(i)$ are obtained from approximation of the diffusion and degradation elements in the PDE. In addition, we define the functional rate for action $mem(i)$ as a the constant value 10. In particular:

$$\begin{aligned} f_{t(i,j)} &= D * M(i) * h/(deltaX * deltaX * h) \\ f_{deg(i)} &= alpha * M(i) * h/h \\ f_{mem(i)} &= 10 \end{aligned}$$

Table 7.1: In this table we illustrate the commitments of the 20 regions of the French Flag Model over 100 simulations and at different time points. For each region, counts over the simulations of commitments (A, B or none, i.e. not committed) are given.

Time	Comm.	1	2	3	4	5	6	7	8	9	10
0s	A	100	0	0	0	0	0	0	0	0	0
	B	0	0	0	0	0	0	0	0	0	0
	none	0	100	100	100	100	100	100	100	100	100
1.5s	A	100	100	99	98	93	81	66	35	17	8
	B	0	0	1	2	7	15	17	30	35	51
	none	0	0	0	0	0	4	17	35	48	41
3s	A	100	100	99	98	93	85	81	62	42	12
	B	0	0	1	2	7	15	17	33	42	72
	none	0	0	0	0	0	0	2	5	16	16
4.5s	A	100	100	99	98	93	85	83	65	47	17
	B	0	0	1	2	7	15	17	33	46	78
	none	0	0	0	0	0	0	0	2	7	5
6s	A	100	100	99	98	93	85	83	66	53	18
	B	0	0	1	2	7	15	17	33	46	79
	none	0	0	0	0	0	0	0	1	1	3

Time	Comm.	11	12	13	14	15	16	17	18	19	20
0s	A	0	0	0	0	0	0	0	0	0	0
	B	0	0	0	0	0	0	0	0	0	0
	none	100	100	100	100	100	100	100	100	100	100
1.5s	A	6	1	0	0	0	0	0	0	0	0
	B	34	38	28	16	10	5	0	0	0	0
	none	60	61	72	84	90	95	100	100	100	100
3s	A	12	3	0	0	0	0	0	0	0	0
	B	73	76	69	48	28	13	2	2	0	0
	none	15	21	31	52	72	87	98	98	100	100
4.5s	A	13	4	0	0	0	0	0	0	0	0
	B	82	88	84	67	48	23	8	2	0	0
	none	5	8	16	33	52	77	92	98	100	100
6s	A	14	4	0	0	0	0	0	0	0	0
	B	83	95	96	79	61	31	12	3	0	0
	none	3	1	4	21	39	69	88	97	100	100

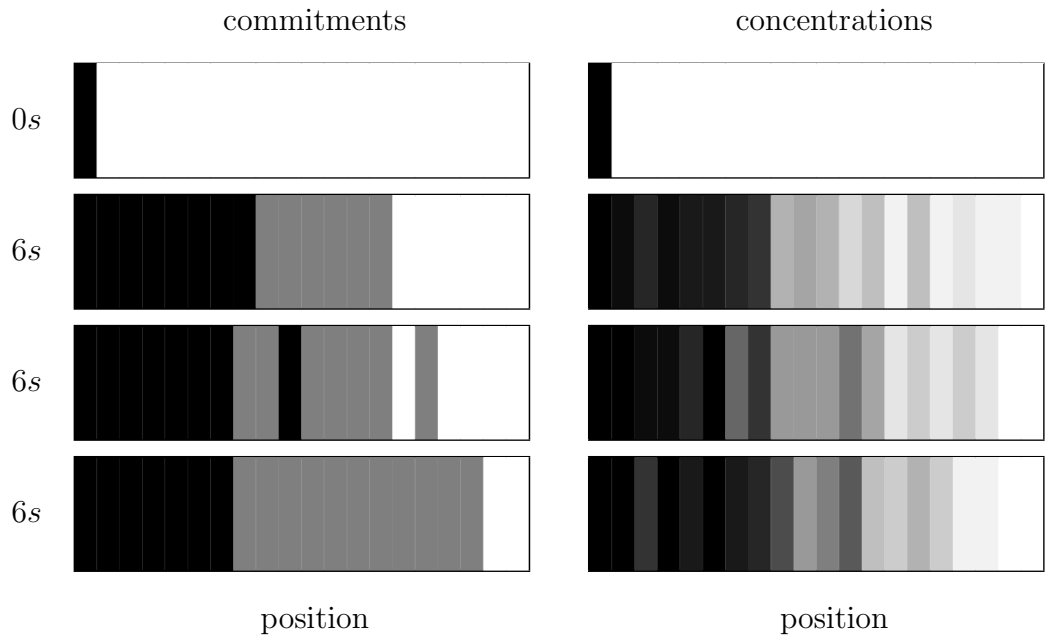


Figure 7.5: Example of simulations of the process algebra with hooks French Flag Model. On the left: commitments of regions to cell specialisations after 6 seconds. On the right: concentration levels after 6 seconds of the same simulation runs. Top row is the initial condition.

7.2.1 Analysis

100 simulations were performed on the model, recording the concentration level of **M** in the three regions, up to 6 seconds. Commitments of regions was also recorded. Figure 7.5 illustrates the initial condition and the typical results from single simulations at time 6 seconds. Although some variability between the runs is visible, a pattern of three distinct commitments is always visible. The images in the figure are constructed from the model component representing the current state at time 6 seconds. Each picture represents a one-dimensional space divided into 20 regions with source of morphogen **M** in the left most region. On the left, commitments to cell differentiation are shown: regions committed to A are represented by the colour black, regions committed to B by grey, while non-committed regions by white. On the right, the corresponding concentration levels are shown: each concentration level is represented by a different shade of grey, from black (maximum concentration) to white (absence of concentration). The top row shows the initial condition (time $0s$), while the other three rows show three different simulations at time $6s$.

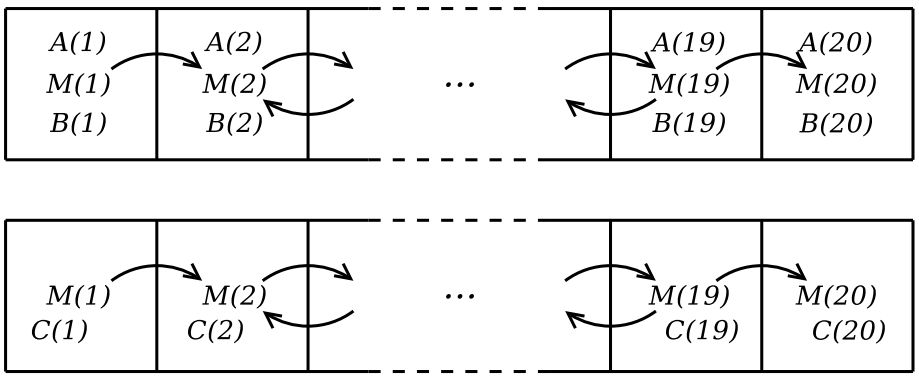


Figure 7.6: Two extensions of the psPAH French Flag Model. In the first extension, top, two species **A** and **B** are added. In the second extension, bottom, species **C** is added.

Additional data is illustrated in Table 7.1. In this table commitments of regions are shown. At time 0, only region 1, the source, is committed to specialisation A, while all the other regions are not committed. As the time approaches 6 seconds, we can see the proportion of the 100 simulations in which the regions commit to a certain specialisation. For example, regions 2 to 5 present a clear preference for specialisation A, while regions 12 and 13 have a marked preference for specialisation B. Although some variability is present, a change in preference from left to right is evident.

7.2.2 Example of Use of Congruence

We illustrate now how the concepts of compositionality and congruence in process algebra can be used to reason about the behaviour of the French Flag Model. In particular, we prove two different extensions of the French Flag model to be Markovian $(\mathfrak{T}, \Gamma)$-bisimilar by extending the equality of part of the system to the equality of the whole system. The two extensions consist of the addition of biochemical species **A** and **B** in the first case and **C** in the second. These new species do not interact with morphogen **M** and do not diffuse, but nevertheless produce their own behaviour becoming part of the system. The two extensions are illustrated in Figure 7.6.

Consider the following two molecular models:

1. the concentration of two molecules **A** and **B** are modelled by agent processes representing their concentration as either high or low. Species **A** and **B** are

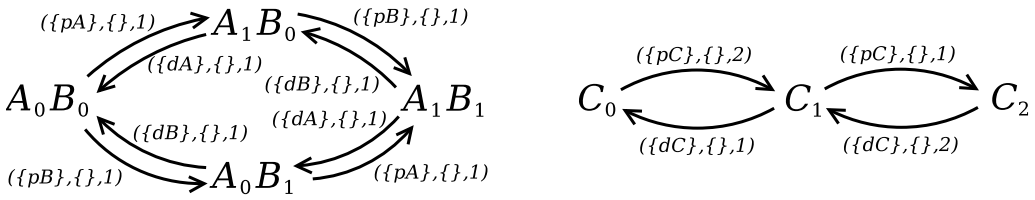


Figure 7.7: Rated derivation graphs of model processes $A_0(n) \underset{\emptyset}{\bowtie} B_0(n)$ and $C_0(n)$, with $n \in \mathbb{R}$. Parameter n is omitted.

produced or degraded independently according to the mass action kinetic law. Process definitions are as follows:

$$A_0(i) \triangleq pA(i).A_1(i) \quad A_1(i) \triangleq dA(i).A_0(i)$$
$$B_0(i) \triangleq pB(i).B_1(i) \quad B_1(i) \triangleq dB(i).B_0(i)$$

Reactions, functional rates and set of participants are as follows (parameters $k = 1$ and $h = 1$):

$$\begin{array}{lll} R_{pA} : \rightarrow \mathbf{A} & f_{pA(i)} = k/h & p_{pA(i)} = \{A(i)\} \\ R_{dA} : \mathbf{A} \rightarrow & f_{dA(i)} = k * A(i) * h/h & p_{dA(i)} = \{A(i)\} \\ R_{pB} : \rightarrow \mathbf{B} & f_{pB(i)} = k/h & p_{pB(i)} = \{B(i)\} \\ R_{dB} : \mathbf{B} \rightarrow & f_{dB(i)} = k * B(i) * h/h & p_{dB(i)} = \{B(i)\} \end{array}$$

The rated derivation graph $\mathcal{D}_r(A_0(n) \underset{\emptyset}{\bowtie} B_0(n))$, where $n \in \mathbb{R}$, is shown in Figure 7.7, on the left.

2. concentration of molecule **C** is modelled with three agents, representing high, medium and low concentration. Species **C** inhibits its own production, so the functional rate associated to the production of **C** is inversely proportional to its concentration. Species **C** can also degrade according to mass action. Process definitions are as follows:

$$C_0(i) \triangleq pC(i).C_1(i) \qquad C_1(i) \triangleq pC(i).C_2(i) + dC(i).C_0(i)$$
$$C_2(i) \triangleq dC(i).C_1(i)$$

Reactions, functional rates and set of participants are as follows (parameters $k = 1$, $k' = 0.5$ and $h = 1$):

$$R_{pC} : \rightarrow \mathbf{C} \quad f_{pC(i)} = 1/(k' * h * (1 + C(i) * h)) \quad p_{pC(i)} = \{C(i)\}$$
$$R_{dC} : \mathbf{C} \rightarrow \quad f_{dC(i)} = k * C(i) * h/h \quad p_{dC(i)} = \{C(i)\}$$

The rated derivation graph $\mathcal{D}_r(C_0(n))$, where $n \in \mathbb{R}$, is shown in Figure 7.7, on the right.

Notice that for all $n \in \mathbb{R}$, $A_0(n) \underset{\emptyset}{\bowtie} B_0(n) \simeq_{\mathcal{T},\Gamma} C_0(n)$ for any $\mathcal{T}$ such that $\mathcal{T} \cap \{\!|pA, pB, pC, dA, dB, dC|\!\} = \emptyset$ and with the appropriate environment Γ. In other words, the two psPAH model components are equivalent if we abstract away from the specific actions they can perform, retaining only the timing and likelihood of those actions. Recall that set $\mathcal{T}$ indicates on which actions rated derivation graphs $\mathcal{D}_r(A_0(n) \underset{\emptyset}{\bowtie} B_0(n))$ and $\mathcal{D}_r(C_0(n))$ should be compared. Moreover, a suitable environment Γ is a set of parameters where constant parameters of the two models are merged without conflict of names.

Assume now that the French Flag Model is updated with the addition of chemicals **A** and **B**, which do not interact with morphogen **M**, but that are nevertheless present in the system. Assume also that we are interested in comparing the behaviour of the resulting model with the behaviour of the French Flag Model updated with the addition of **C** instead. Without the need for looking at the actual behaviour of the two new systems, we can prove that their overall behaviour is identical. We prove this simply using the fact that $A_0(n) \underset{\emptyset}{\bowtie} B_0(n)$ and $C_0(n)$ are Markovian $(\mathcal{T}, \Gamma)$-bisimilar and the fact that Markovian $(\mathcal{T}, \Gamma)$-bisimulation is a congruence (Proposition 6.34) and it is transitive (Proposition 6.28).

Because of Proposition 6.34 we have that:

$$(A_0(1) \underset{\emptyset}{\bowtie} B_0(1)) \underset{\emptyset}{\bowtie} M(1, 20) \simeq_{\mathcal{T},\Gamma} C_0(1) \underset{\emptyset}{\bowtie} M(1, 20)$$

which in turn implies that:

$$\begin{gathered}((A_0(1) \underset{\emptyset}{\bowtie} B_0(1)) \underset{\emptyset}{\bowtie} M(1, 20)) \underset{\{\!|x(1),y(1)|\!\}}{\bowtie} TA(1) \simeq_{\mathcal{T},\Gamma} \\ (C_0(1) \underset{\emptyset}{\bowtie} M(1, 20)) \underset{\{\!|x(1),y(1)|\!\}}{\bowtie} TA(1)\end{gathered} \tag{7.1}$$

In the same way we have:

$$\begin{array}{c}((A_0(2) \underset{\emptyset}{\bowtie} B_0(2)) \underset{\emptyset}{\bowtie} M(2,20)) \underset{\{\!|x(2),y(2)|\!\}}{\bowtie} T(2,0,0) \simeq_{\mathcal{T},\Gamma} \\ (C_0(2) \underset{\emptyset}{\bowtie} M(2,0)) \underset{\{\!|x(2),y(2)|\!\}}{\bowtie} T(2,0,0)\end{array} \tag{7.2}$$

At this point, notice that whenever $X \simeq_{\mathcal{T},\Gamma} Y$ and $W \simeq_{\mathcal{T},\Gamma} Z$, then $X \underset{\mathcal{L}}{\bowtie} W \simeq_{\mathcal{T},\Gamma} X \underset{\mathcal{L}}{\bowtie} Z \simeq_{\mathcal{T},\Gamma} Y \underset{\mathcal{L}}{\bowtie} Z$. By the transitivity of $\simeq_{\mathcal{T},\Gamma}$ (Proposition 6.28), we have $X \underset{\mathcal{L}}{\bowtie} W \simeq_{\mathcal{T},\Gamma} Y \underset{\mathcal{L}}{\bowtie} Z$. Using this result and Equations (7.1) and (7.2) we have:

$$\begin{array}{c}((A_0(1) \underset{\emptyset}{\bowtie} B_0(1)) \underset{\emptyset}{\bowtie} M(1,20)) \underset{\{\!|x(1),y(1)|\!\}}{\bowtie} TA(1) \underset{\{\!|t(1,2)|\!\}}{\bowtie} \\ ((A_0(2) \underset{\emptyset}{\bowtie} B_0(2)) \underset{\emptyset}{\bowtie} M(2,0)) \underset{\{\!|x(2),y(2)|\!\}}{\bowtie} T(2,0,0) \\ \simeq_{\mathcal{T},\Gamma} \\ (C_0(1) \underset{\emptyset}{\bowtie} M(1,20)) \underset{\{\!|x(1),y(1)|\!\}}{\bowtie} TA(1) \underset{\{\!|t(1,2)|\!\}}{\bowtie} \\ (C_0(2) \underset{\emptyset}{\bowtie} M(2,0)) \underset{\{\!|x(2),y(2)|\!\}}{\bowtie} T(2,0,0)\end{array}$$

Continuing this demonstration with the composition of the processes representing the remaining spatial regions we obtain:

$$\begin{array}{c}((A_0(1) \underset{\emptyset}{\bowtie} B_0(1)) \underset{\emptyset}{\bowtie} M(1,20)) \underset{\{\!|x(1),y(1)|\!\}}{\bowtie} TA(1) \underset{\{\!|t(1,2)|\!\}}{\bowtie} \\ ((A_0(2) \underset{\emptyset}{\bowtie} B_0(2)) \underset{\emptyset}{\bowtie} M(2,0)) \underset{\{\!|x(2),y(2)|\!\}}{\bowtie} T(2,0,0) \underset{\{\!|t(2,3)t(3,2)|\!\}}{\bowtie} \\ \cdots ((A_0(20) \underset{\emptyset}{\bowtie} B_0(20)) \underset{\emptyset}{\bowtie} M(20,0)) \underset{\{\!|x(20),y(20)|\!\}}{\bowtie} T(20,0,0) \\ \simeq_{\mathcal{T},\Gamma} \\ (C_0(1) \underset{\emptyset}{\bowtie} M(1,20)) \underset{\{\!|x(1),y(1)|\!\}}{\bowtie} TA(1) \underset{\{\!|t(1,2)|\!\}}{\bowtie} \\ (C_0(2) \underset{\emptyset}{\bowtie} M(2,0)) \underset{\{\!|x(2),y(2)|\!\}}{\bowtie} T(2,0,0) \underset{\{\!|t(2,3)t(3,2)|\!\}}{\bowtie} \\ \cdots (C_0(20) \underset{\emptyset}{\bowtie} M(20,0)) \underset{\{\!|x(20),y(20)|\!\}}{\bowtie} T(20,0,0)\end{array} \tag{7.3}$$

Equation (7.3) finally proves that the addition of molecules **A** and **B** and the addition of molecule **C** to the French Flag Model have the same effect on its spatio-temporal behaviour. The two extended versions of the French Flag Model are Markovian $(\mathcal{T},\Gamma)$-bisimilar for any action set $\mathcal{T}$ as long as $\mathcal{T}$ does not contain actions of molecules **A**, **B** or **C** and Γ is the union of all constant parameters of the different parts composing the model, without conflicts of names. If this is the case we can assert that, for example, the commitment of the regions to a

specialisation ($\mathcal{T} = \{\!|mem(1), \ldots, mem(20)|\!\}$) happens with the same timing and with the same probability in both models.

7.3 Multi-Scale Model of Tissue Growth

We now turn our attention to a multi-scale model of tissue growth. This model is constructed in analogy with our earlier tissue growth model introduced in Section 4.2.3 and modelled in process algebra with hooks in Section 5.1.3. We define it as a multi-scale model because two scales are present, tissue and biochemistry, and the two scales need to interact. In particular, we show how one can define a model where growth and death of tissue depend on the local concentration of biochemical species. Here we give an overview of the model, while leaving the complete specification in Appendix B.1.

At the tissue scale we consider an area divided into regions of the same size and shape. We consider a grid of 10×10 regions, each region is denoted by $R(i, j)$. Each region can be empty (agent $E(i, j)$) or contain tissue. There are four types of tissue: tissue that can neither grow nor die (agent $T(i, j)$); tissue that can grow, but not die ($Tm(i, j)$); tissue that can die but not grow ($Ta(i, j)$); tissue that can both grow and die ($Tam(i, j)$). Tissue processes change between these four agents depending on the configuration of the biochemical scale. The event of growth is represented by action $growth(i, j, i2, j2)$ which is performed by a tissue agent in region $R(i, j)$ in synchronisation with an adjacent empty space $E(i2, j2)$ in region $R(i2, j2)$. Two regions are considered adjacent if they share an edge. If no adjacent region is empty, growth is inhibited. The event of tissue death is represented by action $death(i, j)$. We assume that actions $growth(i, j, i2, j2)$ and $death(i, j)$ have constant rates k_{growth} and k_{death}.

The biochemical scale consists of biochemical species **A**, **B** and **C**, present in all regions. The concentration of each species varies between a concentration level of 0 and 10 (parameter $maxLevels = 10$). In particular, we use agent $A(i, j, w)$ to denote that species **A** in region $R(i, j)$ presents concentration level w. Analogously for species **B** and **C**. Concentration level of the three species can change because of the following local biochemical reactions (and associated velocities):

$$R_1 : \mathbf{A} + \mathbf{B} \rightarrow \mathbf{C} \qquad v_1 = k_1[\mathbf{A}][\mathbf{B}]$$
$$R_2 : \rightarrow \mathbf{A} \quad v_2 = k_2 \qquad R_3 : \rightarrow \mathbf{B} \qquad v_3 = k_3$$
$$R_4 : \mathbf{C} \rightarrow \quad v_4 = k_4[\mathbf{C}] \quad R_5 : \mathbf{B} \rightarrow \mathbf{B'} \quad v_5 = k_5[\mathbf{C}]$$

where R_5 is the transport of concentration of **B** from a compartment to an adjacent compartment. The following constraints, which require communication *between* scales, must hold:

- tissue can grow if and only if the concentration level of **A** in the same region is 5 or more. Actions $growthon(i, j)$ and $growthoff(i, j)$ are used as hook actions to indicate that a threshold has passed at the biochemical scale. Tissue processes can synchronise with these hook actions and change accordingly;
- tissue can die if and only if the concentration level of **C** in the same region is 5 or more (parameter $thr = 5$). Actions $deathon(i, j)$ and $deathoff(i, j)$ are used as hook actions to indicate that a threshold has passed;
- a region is empty if and only if there is no biochemistry. To represent the absence of biochemistry we use processes $NA(i, j)$, $NB(i, j)$ and $NC(i, j)$. Actions $bioon(i, j)$ and $biooff(i, j)$ work across scales and ensure this is the case.

Consider for example the definition of agents $C(i, j, w)$ and $T(i, j)$, shown in Figure 7.8. In a region $R(i, j)$, if the concentration level of **C** (i.e. w) is below its maximum and **A** and **B** are available then **C** can participate in reaction R_1, represented by action $r1(i, j)$. If w is equal to 4 ($w == thr - 1$), then action $r1(i, j)$ carries also hook action $deathon(i, j)$, which in turn could synchronise with the tissue scale, bringing $T(i, j)$ to $Ta(i, j)$. We note that without the use of a parametric version of PAH, 100 definitions of $T(i, j)$ and 1000 definitions of $C(i, j, w)$ processes would have been necessary, one for each region and level of concentration to model.

The complete definition of the model along with parameter values can be found in Appendix B.1. The initial state is a grid composed of agents $E(i, j)$ in all regions with the exception of $R(6, 6)$, where agent $Tm(6, 6)$ is used. At the biochemical scale, agents $NA(i, j)$, $NB(i, j)$ and $NC(i, j)$ are used with the exception of $A(6, 6, 5)$, $B(6, 6, 0)$ and $C(6, 6, 0)$. In terms of model processes,

$C(i,j,w) \triangleq$
$bioff(i,j).NC(i,j)$
$+$
if $w < maxLevels$ **then**
  if $w == (thr-1)$ **then**
    $r1(i,j)[deathon(i,j)].C(i,j,w+1)$
  else
    $r1(i,j).C(i,j,w+1)$
else
  nil
$+$
if $w > 0$ **then**
  if $w == thr$ **then**
    $r4(i,j)[deathoff(i,j)].C(i,j,w-1)$
  else
    $r4(i,j).C(i,j,w-1)$
else
  nil

$T(i,j) \triangleq growthon(i,j).Tm(i,j) + deathon(i,j).Ta(i,j)$

Figure 7.8: Agent definitions of processes $C(i,j,w)$ and $T(i,j)$, from the multi-scale model of tissue growth.

the initial state consists of a vertical synchronisation between a model process representing the entire biochemical scale and the model process representing the tissue scale, $Biochem \underset{\mathcal{H}}{\boxtimes} Tissue$, with cooperation set $\mathcal{H}$ containing the list of all hook actions used in the model.

7.3.1 Analysis

Examples of simulations of the model are shown in Figure 7.9. A region is white when an agent of the type $E(i,j)$ is found, black otherwise (i.e. tissue of some kind is found). As an example of analysis of the system, we focus on the role the production of species **B** has on tissue growth and death. Although **B** does not regulate tissue processes directly, it is involved along with **A** and **C** in reaction R_1. Intuitively, if the concentration of **B** is low, **A** is not consumed and growth

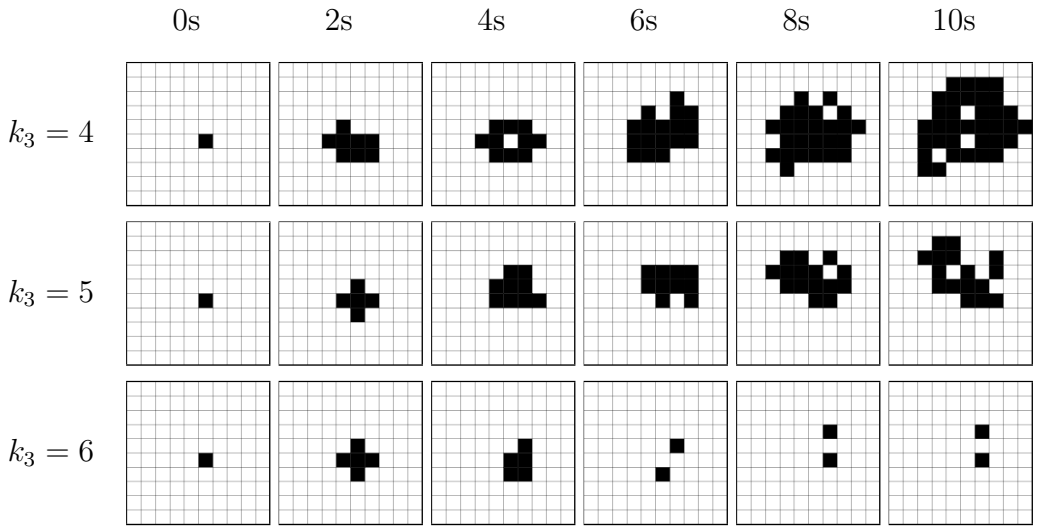


Figure 7.9: Three sample runs with k_3 equal to 4, 5 and 6 $Molar/s$. Black squares represent regions containing tissue.

becomes likely, while **C** is not produced and tissue death becomes unlikely. The parameter which regulates the production of **B** is k_3. Thus, we observed the behaviour of the system using three different values for k_3, 4, 5 and 6 M/s, performing 100 simulation runs for each configuration. The results are shown in Figure 7.10, where one can see that increasing the production rate of **B** decreases the growth/death ratio.

7.4 Discussion

The two multi-scale models presented in this chapter allowed us to give a complete view of how psPAH is intended to be used. In particular, we have seen that:

- psPAH models can be built from existing traditional models. In Section 7.2 we converted a reaction-diffusion PDE model into a psPAH description via a discretisation of space and concentration. In general, this approach introduces uncertainty in the model, because PDEs represent only the most likely behaviour, while psPAH introduces likelihood of events. Uncertainty can be reduced with a finer discretisation at the price of an increase of the number of states of the model. Notice that a finer discretisation only affects

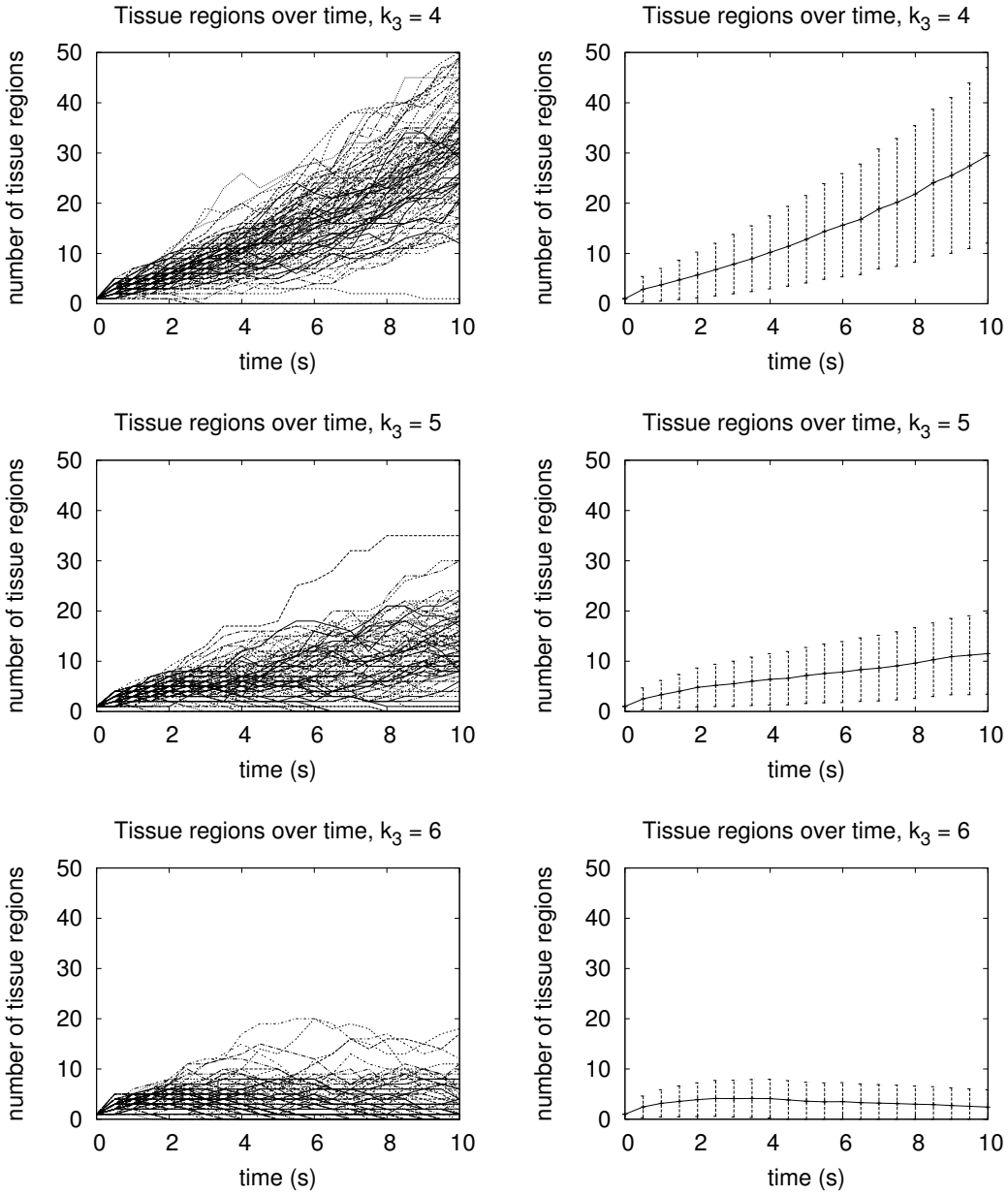


Figure 7.10: Number of tissue regions with parameter k_3 equal to 4, 5 and 6 $Molar/s$, with 100 simulations for each configuration. In the top row, all 100 simulations are shown, while in the bottom row average and standard deviation of the same runs.

the size of the initial state of the psPAH model, while the parametric agent definitions are unaltered;

- additional scales can be easily added to existing models using process algebra compositionality. In Section 7.2, we converted the biochemical scale model of diffusion and degradation of morphogen **M** into a psPAH description and subsequently augmented it with explicit representations of tissue and commitment to tissue specialisations;

- psPAH forces an explicit representation of scales and actions between and within scales, resulting in an explicit multi-scale compositionality. In both models of pattern formation and tissue growth, processes on different scales are clearly distinguishable because of the use of the $\bowtie_{\mathcal{L}}$ operator. While in the first model we distributed $\bowtie_{\mathcal{L}}$ over the spatial regions, in the second model we used $\bowtie_{\mathcal{L}}$ only once, defining a clear distinction between the biochemical and tissue scales. This latter approach guarantees a multi-scale compositionality of the model, where a scale can be easily substituted with another. Notice that this approach is also possible in the first model, though an additional layer is required to manage multiple hook actions ($t(i, j)$ may offer both $y(i)$ and $x(j)$ at the same time);

- a psPAH model can be analysed with traditional techniques. We analysed the behaviour of both models quantitatively using stochastic simulations. An example of sensitivity analysis was also given in the tissue growth example, where a constant parameter was altered at the biochemical scale and change in quantitative behaviour was observed at the tissue scale;

- qualitative information about quantitative behaviour can be derived using process algebra related techniques. In Section 7.2.2 we proved that two extensions of the French Flag Model have the same spatio-temporal behaviour, without investigating their behaviour quantitatively. Moreover, this was achieved by simply observing that just a small part of the two models was equivalent and then proving that this equivalence could be extended to the entire models.

7.5 Summary

In this section we have illustrated the use of process algebra with hooks to define and simulate multi scale models of pattern formation and tissue growth. In order to define the models compactly we extended the algebra with parametric processes and actions and with the if-then-else construct. We illustrated how parametric stochastic process algebra with hooks can be used to model and analyse two multi-scale scenarios: pattern formation and tissue growth. Finally, we highlighted the benefits of using a process algebraic approach in these scenarios. Most notably, we have seen how one can manipulate models and reason on their behaviour using process algebra with hooks compositionality and theoretical results.

Chapter 8

Conclusions

The multi-scale modelling of biological systems consists of the integration in the same model of multiple levels of detail, from molecules, to cells, to tissues and so on. Usually, different levels of detail are represented by different mathematical approaches, e.g. concentration of molecules by ordinary differential equations, evolution of tissue by cellular automata. Moreover, models are often tailored around the specific system under study. As a result, multi-scale models are difficult to write, maintain, share, compose and compare.

In this thesis we have demonstrated that process algebra, and in particular process algebra with hooks, provides an effective mathematical framework to construct, compose and compare multi-scale models.

First, we investigated the use of a simple process algebra to model a single scale of a biological phenomenon, focussing on biochemical reactions and tissue growth. As a result of this investigation, we proposed a novel approach to functional rates in process algebra with multi-way synchronisation, where we assume that actions can be rated only when all the expected participants synchronise. In addition, we considered the use of parameters to reduce the length of definitions in models with repetitive process definition, which is often the case if concentration levels and geometrical space are considered.

Second, we investigated the use of the simple process algebra and a process algebra with priorities to model multi-scale scenarios, with attention to the interactions between scales and the representation of thresholds. In particular, we assumed that the quantitative behaviour at a scale affects other scales only when certain configurations or states are reached, for example when one or more bio-

chemical species passes a concentration threshold. We observed that the simple process algebra encountered a combinatorial problem in the process definition when multiple thresholds are considered. Conversely, the process algebra with priority showed to possess mechanisms able to represent effectively both interactions within and between scales. However, we demonstrated that the style of modelling of process algebra with priorities presented drawbacks which, in a multi-scale scenario, require to be addressed. The drawbacks are: the introduction of intermediate and biologically meaningless states; the lack of control by the modeller on the response to simultaneous inter-scale events; the lack of syntactic elements that can define unambiguously scales and interactions within and between scales.

Third, we proposed process algebra with hooks for multi-scale modelling of biological systems. Building up from our investigations, we defined this algebra to address the drawbacks of process algebra with priorities and to support our approach to functional rates. Characteristics of the algebra are the use of composed actions and a novel vertical composition operator. In addition, we defined three congruence relations on process algebra with hooks. In particular, one of them, Markovian $(\mathcal{T}, \Gamma)$-bisimulation, has been designed to relate models that present the same behaviour at a specified scale, at the same time with the same probability. Most notably, our approach to functional rates was fundamental to the proof of congruence.

Finally, we illustrated the use of process algebra with hooks to define, compose and relate models of pattern formation and tissue growth.

With the above results we demonstrated that:

- **define.** Process algebra with hooks can be used independently of the scale one decides to model first. Each scale can be treated as the same formal object: a process. For example, we modelled the biochemical scale (Section 5.2.3) or the tissue scale (Section 5.2.4);

- **compose.** Composition of scales is facilitated by the intrinsic compositionality of the process algebra approach and by the novel vertical cooperation operator. Interactions between scales are unambiguously identified by hook actions. For example, we modelled interactions between the biochemical scale and the tissue scale in Sections 7.2 and 7.3;

- **relate.** The framework of process algebra with hooks allows formal reasoning about behaviour of models. As a consequence, relations can be defined between models, capturing a specified degree of equality or similarity. In particular, we defined three equivalence relations in Sections 6.2.1 and 6.2.2 and we illustrated how one of them can be used to infer qualitative information about quantitative behaviour in Section 7.2.2.

Although we have set the foundations of process algebra with hooks for multi-scale models, further research is necessary before it can be considered as competitive as other more consolidated mathematical approaches. At present, three challenges need to be addressed. First, an efficient stochastic simulation algorithm needs to be defined, in order to allow modelling of large systems and understand the time and space complexity limitations of using process algebra with hooks. Second, other equivalence relations should be defined. The ability to relate models and their behaviour is the most appealing feature of the process algebra approach. Although we defined three fundamental relations, these may be too strong to relate biological systems, where often one is interested in *how similar* two models are, rather than whether they are *exactly* the same. The time complexity of testing of the equalities has also to be investigated. Finally, process algebra with hooks needs to be compared with state of the art approaches to multi-scale modelling. This will be possible only once the two previous points are addressed. Without efficient simulations and flexible equivalence relations it would be difficult to quantify differences in time complexity and model manipulation.

In the following section we outline how one should proceed to address the above lines of research.

8.1 Future Directions

- *efficient stochastic simulations of process algebra with hooks models.* At the moment, the simulator that samples trajectories of transitions is not optimised. In fact, at each step the complete set of rated transitions is computed, while it could be the case that only a few processes are affected by the last step and that only a few transitions need to be updated. A first improvement is to determine which processes are affected by a transition.

With this information, improvements to stochastic simulations of biochemical reactions and their diffusion, e.g. (Elf and Ehrenberg, 2004; Gibson and Bruck, 2000), can be adapted and integrated in our approach;

- *definition of approximate relations on process algebra with hooks processes.* We may require a more qualitative interpretation of behaviour, where two systems can be considered similar, though not identical. Approximate equivalence relations (Tini, 2010) and distance measures between models may prove an interesting direction to explore if we want to compare biological systems. Parameters that identify positions could be used to define metrics for approximate equivalence relations;

- *Comparison of process algebra with hooks with other approaches.* The above two points will have to be developed in order to have a fair comparison. Consider, for example, the case of comparing process algebra with hooks with cellular automata. Fundamental to cellular automata is the ability to provide fast simulations of large systems, while it does not provide the mathematical framework of process algebra. In order to compare the two approaches, one needs to determine how the time complexity of optimised simulations of process algebra with hooks models compares with that of cellular automata, and what flexible relations on processes provide a practical advantage.

8.2 Summary

We have developed and illustrated the use of a novel process algebra designed for multi-scale modelling of biological systems. Its mathematical framework provides a flexible environment where quantitative, dynamic, multi-scale models of biological systems can be defined, composed and compared.

We have provided the foundations of a new approach; future work is required to determine its full potential.

Appendix A

Stochastic Process Algebra with Priorities

In this appendix we define a stochastic semantics for PAwP, in analogy with Section 3.2.1.

Recall Example 2 of Section 4.2.1:

$$A_0 \triangleq 1{:}a.A_1 + 1{:}f.A_1 \qquad A_1 \triangleq 1{:}a.A_2 + 1{:}f.A_2 + 1{:}b.A_0 + 1{:}e.A_0 \qquad A_2 \triangleq 1{:}b.A_1 + 1{:}e.A_1$$

$$B_0 \triangleq 1{:}c.B_1 + 1{:}e.B_1 \qquad B_1 \triangleq 1{:}c.B_1' + 1{:}e.B_1' + 1{:}d.B_0 + 1{:}f.B_0 \qquad B_2 \triangleq 1{:}d.B_2' + 1{:}f.B_2'$$

$$B_1' \triangleq 2{:}x.B_2 \qquad B_2' \triangleq 2{:}y.B_1$$

$$Cell_M \triangleq 1{:}move.Cell_M + 2{:}x.Cell_A$$

$$Cell_A \triangleq 1{:}absorb.Cell_A + 2{:}y.Cell_M$$

The initial state is:

$$(A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M$$

An appropriate stochastic semantics for PAwP based on functional rates should construct an environment Γ to evaluate functional rates when the action considered has priority 1. This should be done in analogy with our approach for sSPA in Section 3.2.1. If the action considered has a priority higher than 1, then the rate should be ∞, that is the action has no delay and is preformed instantaneously.

As in Section 3.2.1 we assign variables and values to agents:

$$
\begin{array}{llll}
Var(A_0) = A & Var(B_0) = B & Var(B'_1) = B & Var(Cell_M) = cell \\
Val(A_0) = 0 & Val(B_0) = 0 & Val(B'_1) = 2 & Val(Cell_M) = 1 \\
Var(A_1) = A & Var(B_1) = B & Var(B'_2) = B & Var(Cell_A) = cell \\
Val(A_1) = 1 & Val(B_1) = 1 & Val(B'_2) = 1 & Val(Cell_A) = 2 \\
Var(A_2) = A & Var(B_2) = B & & \\
Val(A_2) = 2 & Val(B_2) = 2 & &
\end{array}
$$

Functional rates and sets of participants for actions are:

$$
\begin{array}{llll}
f_a = k_a/h & p_a = \{A\} & f_f = (k_f * B * h)/h & p_f = \{A, B\} \\
f_b = (k_b * A * h)/h & p_b = \{A\} & f_{move} = k_{move} & p_{move} = \{cell\} \\
f_c = k_c/h & p_c = \{B\} & f_{absorb} = k_{absorb} & p_{absorb} = \{cell\} \\
f_d = (k_d * B * h)/h & p_d = \{B\} & f_x = \infty & p_x = \{B, cell\} \\
f_e = (k_e * A * h)/h & p_e = \{A, B\} & f_y = \infty & p_y = \{B, cell\}
\end{array}
$$

A valid derivation for the above example should be:

$$(A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M \xrightarrow{(1:b,\Gamma)} (A_0 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M$$

where $\Gamma = \{(A, 1)\}$. From state $(A_1 \underset{\{e,f\}}{\bowtie} B'_1) \underset{\{x,y\}}{\bowtie} Cell_M$ the only derivation possible should be:

$$(A_1 \underset{\{e,f\}}{\bowtie} B'_1) \underset{\{x,y\}}{\bowtie} Cell_M \xrightarrow{(2:x,\Gamma')} (A_1 \underset{\{e,f\}}{\bowtie} B_2) \underset{\{x,y\}}{\bowtie} Cell_A$$

where $\Gamma' = \{(B, 2), (cell, 1)\}$. Although Γ' is not used to evaluate f_x, it is necessary to verify that $envVar(\Gamma')= p_x$, i.e. that the variable contained in Γ are exactly the variables contained in the set of participants p_x. The operation of rating should then distinguish between actions with priority 1 and actions with priority higher than 1. This means that valid transitions should carry the priority of the action on their label. Using the additional environment $\Gamma = \{(h, 1), (k_a, 1), (k_b, 1), (k_c, 1), (k_d, 1), (k_e, 1), (k_f, 1), (k_{move}, 1), (k_{absorb}, 1)\}$, two examples of rated transitions should be:

$$(A_1 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M \xrightarrow{(b,1)} (A_0 \underset{\{e,f\}}{\bowtie} B_1) \underset{\{x,y\}}{\bowtie} Cell_M$$
$$(A_1 \underset{\{e,f\}}{\bowtie} B_1') \underset{\{x,y\}}{\bowtie} Cell_M \xrightarrow{(x,\infty)} (A_1 \underset{\{e,f\}}{\bowtie} B_2) \underset{\{x,y\}}{\bowtie} Cell_A$$

where the priority of the actions is now not necessary and so dropped.

Recall that we are assuming that we can always determine all the biological entities that interact for a specific event. This allows us to identify unequivocally which processes will participate to an action in a certain moment. This is our assumption for biological interactions within a scale. However, this might not hold in the case of interactions between scales via instantaneous actions. Consider the following example:

$$A \triangleq 1{:}a.A' \qquad B \triangleq 1{:}b.B' \qquad C \triangleq 2{:}x.C'$$
$$A' \triangleq 2{:}x.A'' \qquad B' \triangleq 2{:}x.B''$$

$$Var(A) = levelA \qquad Var(B) = levelB \qquad Var(C) = celltype$$
$$Var(A') = levelA \qquad Var(B') = levelB$$

with initial state:

$$(A \underset{\emptyset}{\bowtie} B) \underset{\{x\}}{\bowtie} C$$

Assume now that processes A and B represent the biochemical scale and C the cellular scale. In this example, the condition for C to change its state is that either A performs action a *or* B performs action b. This implies that the instantaneous action x can have two different sets of participants, that is $\{levelA, celltype\}$ and $\{levelB, celltype\}$. In order to allow the determination of the correct participants to instantaneous actions, we impose that if $f_x = \infty$ then the set of participants p_x is a set of sets of parameter names. In this case we have:

$$f_a = k_a/h \quad p_a = \{levelA\}$$
$$f_b = k_b/h \quad p_b = \{levelB\}$$
$$f_x = \infty \qquad p_x = \{\{levelA, celltype\}, \{levelB, celltype\}\}$$

Another constraint we have to impose is that given prioritised action $p{:}x$, $f_x = \infty$ if and only if $p > 1$. This way, instantaneous actions are exactly all actions with priority higher than 1.

A.1 Stochastic Process Algebra with Priorities

In analogy with Section 3.2.1 we proceed to the definition of a new syntax and semantics for PAwP. The syntax is given by:

$$D ::= nil \mid p{:}a.A \mid D + D$$

$$M ::= A \mid M \underset{\mathcal{L}}{\bowtie} M$$

where:

- D is a *definition* process, $D \in \mathbb{P}_d$, while M is a *model* process, $M \in \mathbb{P}_m$. Definition and model processes are disjoint and are both processes, i.e. $\mathbb{P}_d \cup \mathbb{P}_m = \mathbb{P}$ and $\mathbb{P}_d \cap \mathbb{P}_m = \emptyset$;
- Agents are defined as $A \triangleq D$, that is we use definition processes to define the behaviour of agents;
- a model is defined by a model process M, which in turn is either an agent A or a cooperation between model processes $M \underset{\mathcal{L}}{\bowtie} M$;
- action execution $p{:}a.A$ is always followed by an agent A. This ensures that at any time the state of a model will be constituted of cooperations of agents;
- functions $Var(A)$ and $Val(A)$ must be defined for each agent A, with $Var(A) \in Names$, $Val(A) \in \mathbb{R}$ and $Names$ the set of parameter names.

In order to define a stochastic semantics for this new syntax, we use the following additional definitions:

- $PM(M)$, $M \in \mathbb{P}_m$, returns the potential moves of a process M, $PM(M) \subseteq (\mathbb{N} \setminus \{0\}) \times Actions \times 2^{Names \times \mathbb{R}} \times \mathbb{P}_m$. $PM(M)$ is defined by structural induction as:

$$PM(nil) = \emptyset$$

$$PM(p{:}a.A) = \{\!|(p, a, A)|\!\}$$

$$PM(D_1 + D_2) = PM(D_1) \uplus PM(D_2)$$

$$PM(A) = \{\!|(p, a, \Gamma, A) \mid (p, a, A) \in PM(D) \wedge A \triangleq D \wedge \Gamma = \{(Var(A), Val(A))\}|\!\}$$

$$\begin{aligned} PM(M_1 \underset{\mathcal{L}}{\bowtie} M_2) = {} & \{\!|(p, a, \Gamma, M_1' \underset{\mathcal{L}}{\bowtie} M_2) \mid (p, a, \Gamma, M_1') \in PM(M_1) \wedge a \notin \mathcal{L}|\!\} \\ & \uplus \{\!|(p, a, \Gamma, M_1 \underset{\mathcal{L}}{\bowtie} M_2') \mid (p, a, \Gamma, M_2') \in PM(M_2) \wedge a \notin \mathcal{L}|\!\} \\ & \uplus \{\!|(p, a, \Gamma_1 \cup \Gamma_2, M_1' \underset{\mathcal{L}}{\bowtie} M_2') \mid (p, a, \Gamma_1, M_1') \in PM(M_1) \\ & \wedge (p, a, \Gamma_2, M_2') \in PM(M_2) \wedge a \in \mathcal{L}|\!\} \end{aligned}$$

- $Select(PM(M))$ returns the potential moves of M with the highest priority and is defined as:

$$Select(PMSet) = \{\!|(p, a, \Gamma, M) \mid (p, a, \Gamma, M) \in PMSet \wedge \forall (q, b, \Gamma', M') \in PMSet.p \geq q|\!\}$$

Using the functions defined above, a stochastic semantics for PAwP of actions and functional rates is given by the following derivation rule:

$$\frac{(p, a, \Gamma, M') \in Select(PM(M))}{M \xrightarrow{(p:a,\Gamma)} M'}$$

We refer to this new process algebra as *stochastic process algebra with priorities* (sPAwP). Most of the definitions we introduced in Section 3.2.1 are almost unchanged. The definitions differ only in the fact that they need to consider priorities and instantaneous actions, with rate ∞.

Definition A.1 *Activity.* The couple $(p{:}a, \Gamma)$ such that $p \in (\mathbb{N} \setminus \{0\})$, $a \in Actions$ and $\Gamma \subseteq Names \times \mathbb{R}$ is called an activity.

Definition A.2 *One step derivative.* If $M \xrightarrow{(p:a,\Gamma)} M'$ then M is a one step derivative of M.

Definition A.3 *Derivative.* If $M_i \xrightarrow{(p:a,\Gamma)} \ldots \xrightarrow{(p':a',\Gamma')} M_j$ then M_j is a derivative of M_i.

Definition A.4 *Derivative Set.* The derivative set of a model process $M \in \mathbb{P}_m$ is denoted by $ds(M)$ and is defined as the smallest set of model processes such that:

- $M \in ds(M)$;
- if $M_i \in ds(M)$ and $M_i \xrightarrow{(p:a,\Gamma)} M_j$ then $M_j \in ds(M)$.

Definition A.5 *Current activities for model Processes.* The set of activities that $M \in \mathbb{P}_m$ can perform is denoted by $Activities(M)$ and is defined as:

$$Activities(M) = \{\!|(p{:}a, \Gamma) \mid (p, a, \Gamma, M') \in Select(PM(M))|\!\}$$

Definition A.6 *Activity set.* The set of all activities that a model process $M \in \mathbb{P}_m$ or one of its derivatives can perform is given by:

$$\overrightarrow{Activities}(M) = \biguplus_{M_i \in ds(M)} Activities(M_i)$$

Definition A.7 *Derivation graph.* Given a model component $M \in \mathbb{P}_m$, the derivation graph $\mathcal{D}(M)$ is the labelled directed graph with:

- set of nodes $ds(M)$;
- multi set of transition labels $\overrightarrow{Activities}(M)$;
- multi set of labelled transitions $\rightarrow \subseteq ds(M) \times \overrightarrow{Activities}(M) \times ds(M)$. Given $M' \in ds(M)$, $(M', p{:}a, \Gamma, M'') \in \rightarrow$ iff $M' \xrightarrow{(p:a,\Gamma)} M''$.

Definition A.8 *Open activity.* An open activity is an activity $(p{:}a, \Gamma)$ where:

- if $p = 1$ then Γ does not contain the *exact* variables present in the participant set p_a, i.e. envVar(Γ)$\neq p_a$;
- if $p \neq 1$ then the set of variables contained in Γ is not in the set of possible sets of participants p_a, i.e. envVar(Γ)$\notin p_a$.

Definition A.9 *Function* openActivities. The function 'openActivities' selects open activities from a set of activities $A \subseteq (\mathbb{N} \setminus \{0\}) \times Actions \times 2^{Names \times \mathbb{R}}$:

$$\text{openActivities}(A) = \{\!|(p{:}a, \Gamma) \mid ((p = 1 \wedge \text{envVar}(\Gamma) \neq p_a) \\ \vee (p \neq 1 \wedge \text{envVar}(\Gamma) \notin p_a)) \wedge (p{:}a, \Gamma) \in A|\!\}$$

Definition A.10 *Current open activities.* Given a model process $M \in \mathbb{P}_m$, the set of open activities that P can perform is defined as:

$$OpenAct(M) = \text{openActivities}(Activities(M))$$

Definition A.11 *Open activity set.* The set of all open activities that a model process $M \in \mathbb{P}_m$ can perform is given by:

$$\overrightarrow{OpenAct}(M) = \text{openActivities}(\overrightarrow{Activities}(M))$$

Definition A.12 *Closed activity.* A closed activity is an activity $(p{:}a, \Gamma)$ where :

- if $p = 1$ then Γ contains the *exact* variables present in the participant set p_a, i.e. envVar(Γ)= p_a;
- if $p \neq 1$ then the set of variables contained in Γ is in the set of possible sets of participants p_a, i.e. envVar(Γ)$\in p_a$.

Definition A.13 *Function* closedActivities. The function 'closedActivities' selects closed activities from a set of activities $A \subseteq (\mathbb{N} \setminus \{0\}) \times Actions \times 2^{Names \times \mathbb{R}}$:

$$\text{closedActivities}(A) = (A \setminus \text{openActivities}(A))$$

Definition A.14 *Current closed activities.* Given a model process $M \in \mathbb{P}_m$, the set of closed activities that M can perform is defined as:

$$ClosedAct(M) = \text{closedActivities}(Activities(M))$$

Definition A.15 *Closed activity set.* The set of all open activities that a model process $M \in \mathbb{P}_m$ can perform is given by:

$$\overrightarrow{ClosedAct}(M) = \text{closedActivities}(\overrightarrow{Activities}(M))$$

Definition A.16 *Rated activity.* The couple (a, r) such that $a \in Actions$ and $r \in \mathbb{R}_{>0} \cup \{\infty\}$ is called a rated activity.

Definition A.17 *Function* rateActivities. Given an environment Γ, "rateActivities" converts a set of activities $A \subseteq (\mathbb{N} \setminus \{0\}) \times Actions \times 2^{Names \times \mathbb{R}}$ into a set of rated activities $B \subseteq Actions \times (\mathbb{R} \cup \{\infty\})$:

$$\begin{gathered}\text{rateActivities}(\Gamma)(A) = \\ \{\!|(a, r_a) \mid ((\Gamma \cup \Gamma' \vdash f_a \to k \wedge r_a = k/\pi(A, (p{:}a, \Gamma')) \wedge p = 1) \\ \vee (p \neq 1 \wedge r_a = \infty)) \wedge \ (p{:}a, \Gamma') \in A \ \wedge \ f_a \in \mathbb{F}|\!\}\end{gathered}$$

where $\pi(A, (p{:}a, \Gamma'))$ returns the number of occurrences of $(p{:}a, \Gamma')$ in the multi set A.

Definition A.18 *Current rated activities.* Given a model process $M \in \mathbb{P}_m$ and an environment $\Gamma \subseteq 2^{Names \times \mathbb{R}}$, the set of rated activities that M can perform is defined as:

$$RatedAct(M)_\Gamma = \text{rateActivities}(\Gamma)(ClosedAct(M))$$

$RatedAct(M)_\Gamma$ can be written $RatedAct(M)$ if Γ is clear from the context.

Definition A.19 *Rated activity set.* Given an environment $\Gamma \subseteq 2^{Names \times \mathbb{R}}$, the set of all rated activities that a model process $M \in \mathbb{P}_m$ can perform is given by:

$$\overrightarrow{RatedAct}(M)_\Gamma = \text{rateActivity}(\Gamma)(\overrightarrow{ClosedAct}(M))$$

$\overrightarrow{RatedAct}(M)_\Gamma$ can be written $\overrightarrow{RatedAct}(M)$ if Γ is clear from the context.

Definition A.20 *Rated derivation graph.* Given a model process $M \in \mathbb{P}_m$ and an environment $\Gamma \subseteq 2^{Names \times \mathbb{R}}$, the rated derivation graph $\mathcal{D}_r(M)_\Gamma$ is the labelled directed graph with:

- set of nodes $ds(M)$;
- multi set of transition labels $\overrightarrow{RatedAct}(M)_\Gamma$;
- multi set of labelled transitions $\rightarrow_r \subseteq ds(M) \times \overrightarrow{RatedAct}(M)_\Gamma \times ds(M)$. Given $M' \in ds(M)$, $(M', a, r_a, M'') \in \rightarrow_r$ iff $M' \xrightarrow{(p:a,\Gamma')} M''$, $(p{:}a, \Gamma') \in ClosedAct(M)$ and $\{\!|(a,k)|\!\} = \text{rateActivities}(\Gamma)(\{\!|(p{:}a,\Gamma')|\!\})$ and $r_a = k/\pi(ClosedAct(M), (p{:}a, \Gamma'))$.
- multi set of labelled transitions $\rightarrow_o \subseteq ds(M) \times \overrightarrow{OpenAct}(M) \times ds(M)$. Given $M' \in ds(M)$, $(M', p{:}a, \Gamma', M'') \in \rightarrow_o$ iff $M' \xrightarrow{(p:a,\Gamma')} M''$ and $(p{:}a, \Gamma') \in OpenAct(M)$.

$\mathcal{D}_r(M)_\Gamma$ can be written $\mathcal{D}_r(M)$ if Γ is clear from the context.

Appendix B

Case Study Complete Model Definitions

B.1 Detailed Definition of the Multi-Scale Model of Tissue Growth

In this section we provide the complete definition of the multi-scale model of tissue growth presented in Section 7.3

Constants:

$k_1 = 1/(Ms)$	$k_2 = 5\ M/s$	$k_3 = 5\ M/s$
$k_4 = 1/s$	$k_5 = 1/s$	$h = 1\ M$
$k_{death} = 1\ event/s$	$k_{growth} = 1\ event/s$	$maxLevels = 10$
$rows = 10$	$cols = 10$	$thr = 5$

Functional rates and sets of participants:

$$
\begin{aligned}
f_{r1(i,j)} &= k_1 * A(i,j) * h * B(i,j) * h/h \\
p_{r1(i,j)} &= \{A(i,j), B(i,j), C(i,j)\} \\
f_{r2(i,j)} &= k_2/h \\
p_{r2(i,j)} &= \{A(i,j)\} \\
f_{r3(i,j)} &= k_3/h
\end{aligned}
$$

$$
\begin{aligned}
p_{r3(i,j)} &= \{B(i,j)\} \\
f_{r4(i,j)} &= k4 * C(i,j) * h/h \\
p_{r4(i,j)} &= \{C(i,j)\} \\
f_{r5(i,j,i2,j2)} &= k5 * B(i,j) * h/h \\
p_{r5(i,j)} &= \{B(i,j), B(i2,j2)\} \\
f_{death(i,j)} &= k_{death} \\
p_{death(i,j)} &= \{R(i,j)\} \\
f_{growth(i,j,i2,j2)} &= k_{growth} \\
p_{growth(i,j,i2,j2)} &= \{R(i,j), R(i2,j2)\}
\end{aligned}
$$

Agent definitions:

$NA(i,j) \triangleq bioon(i,j).A(i,j,0)$

$A(i,j,w) \triangleq biooff(i,j).NA(i,j)$

$+($

if $w < maxLevels$ then

if $w == (thr-1)$ then

$r2(i,j)[growthon(i,j)].A(i,j,w+1)$

else $r2(i,j).A(i,j,w+1)$

else nil)

$+($

if $w > 0$ then

if $w == thr$ then

$r1(i,j)[growthoff(i,j)].A(i,j,w-1)$

else $r1(i,j).A(i,j,w-1)$

else nil)

$NB(i,j) \triangleq bioon(i,j).B(i,j,0)$

$B(i,j,w) \triangleq \mathit{biooff}(i,j).NB(i,j)$
$+($
if $w < maxLevels$ then
$r3(i,j).B(i,j,w+1)$
$+($if $i > 1$ then $r5(i-1,j,i,j).B(i,j,w+1)$ else $nil)$
$+($if $i < rows$ then $r5(i+1,j,i,j).B(i,j,w+1)$ else $nil)$
$+($if $j > 1$ then $r5(i,j-1,i,j).B(i,j,w+1)$ else $nil)$
$+($if $j < cols$ then $r5(i,j+1,i,j).B(i,j,w+1)$ else $nil)$
else $nil)$
$+($
if $w > 0$ then
$r1(i,j).B(i,j,w-1)$
$+($if $i > 1$ then $r5(i,j,i-1,j).B(i,j,w-1)$ else $nil)$
$+($if $i < rows$ then $r5(i,j,i+1,j).B(i,j,w-1)$ else $nil)$
$+($if $j > 1$ then $r5(i,j,i,j-1).B(i,j,w-1)$ else $nil)$
$+($if $j < cols$ then $r5(i,j,i,j+1).B(i,j,w-1)$ else $nil)$
else $nil)$

$NC(i,j) \triangleq \mathit{bioon}(i,j).C(i,j,0)$
$C(i,j,w) \triangleq \mathit{biooff}(i,j).NC(i,j)$
$+($
if $w < maxLevels$ then
if $w == (thr-1)$ then
$r1(i,j)[\mathit{deathon}(i,j)].C(i,j,w+1)$
else $r1(i,j).C(i,j,w+1)$
else $nil)$
$+($
if $w > 0$ then
if $w == thr$ then

$$r4(i,j)[deathoff(i,j)].C(i,j,w-1)$$
$$\text{else } r4(i,j).C(i,j,w-1)$$
$$\text{else } nil)$$

$$E(i,j) \triangleq$$
$$(\text{if } i > 1 \text{ then } growth(i-1,j,i,j)[bioon(i,j)].T(i,j) \text{ else } nil)$$
$$+(\text{if } i < rows \text{ then } growth(i+1,j,i,j)[bioon(i,j)].T(i,j) \text{ else } nil)$$
$$+(\text{if } j > 1 \text{ then } growth(i,j-1,i,j)[bioon(i,j)].T(i,j) \text{ else } nil)$$
$$+(\text{if } j < cols \text{ then } growth(i,j+1,i,j)[bioon(i,j)].T(i,j) \text{ else } nil)$$

$$T(i,j) \triangleq growthon(i,j).Tm(i,j) + deathon(i,j).Ta(i,j)$$
$$Tm(i,j) \triangleq$$
$$(\text{if } i > 1 \text{ then } growth(i,j,i-1,j).Tm(i,j) \text{ else } nil)$$
$$+(\text{if } i < rows \text{ then } growth(i,j,i+1,j).Tm(i,j) \text{ else } nil)$$
$$+(\text{if } j > 1 \text{ then } growth(i,j,i,j-1).Tm(i,j) \text{ else } nil)$$
$$+(\text{if } j < cols \text{ then } growth(i,j,i,j+1).Tm(i,j) \text{ else } nil)$$
$$+growthoff(i,j).T(i,j) + deathon(i,j).Tam(i,j)$$
$$+\{growthoff(i,j), deathon(i,j)\}.Ta(i,j)$$
$$Ta(i,j) \triangleq death(i,j)[biooff(i,j)].E(i,j)$$
$$+apooff(i,j).T(i,j) + mitoon(i,j).Tam(i,j)$$
$$+\{growthon(i,j), deathoff(i,j)\}.Ta(i,j)$$
$$Tam(i,j),[R(i,j),2] \triangleq death(i,j)[biooff(i,j)].E(i,j)$$
$$+(\text{if } i > 1 \text{ then } growth(i,j,i-1,j).Tam(i,j) \text{ else } nil)$$
$$+(\text{if } i < rows \text{ then } growth(i,j,i+1,j).Tam(i,j) \text{ else } nil)$$
$$+(\text{if } j > 1 \text{ then } growth(i,j,i,j-1).Tam(i,j) \text{ else } nil)$$
$$+(\text{if } j < cols \text{ then } growth(i,j,i,j+1).Tam(i,j) \text{ else } nil)$$
$$+growthoff(i,j).Ta(i,j) + deathoff(i,j).Tm(i,j)$$

Associated variables and values:

$$
\begin{aligned}
&Var(NA(i,j)) = Var(A(i,j,w)) = A(i,j);\\
&Val(NA(i,j)) = 0;\ Val(A(i,j,w)) = w;\\
&Var(NB(i,j)) = Var(B(i,j,w)) = B(i,j);\\
&Val(NB(i,j)) = 0;\ Val(B(i,j,w)) = w;\\
&Var(NC(i,j)) = Var(C(i,j,w)) = C(i,j);\\
&Val(NC(i,j)) = 0;\ Val(C(i,j,w)) = w;\\
&Var(E(i,j)) = Var(T(i,j)) = Var(Ta(i,j)) = Var(Tm(i,j))\\
&\quad = Var(Tam(i,j)) = R(i,j);\\
&Val(E(i,j)) = 0;\ Val(T(i,j)) = 1;\ Val(Ta(i,j)) = 1;\\
&Val(Tm(i,j)) = 1;\ Val(Tam(i,j)) = 1;
\end{aligned}
$$

Model process and initial state:

$$
\begin{aligned}
&((NA(1,1) \underset{\mathcal{L}_{1,1}}{\bowtie} NB(1,1) \underset{\mathcal{L}_{1,1}}{\bowtie} NC(1,1)) \underset{\mathcal{K}_{1,1}}{\bowtie}\\
&\dots \underset{\mathcal{K}_{1,9}}{\bowtie} (NA(1,10) \underset{\mathcal{L}_{1,10}}{\bowtie} NB(1,10) \underset{\mathcal{L}_{1,10}}{\bowtie} NC(1,10))\\
&) \underset{\mathcal{K}_{1,10}}{\bowtie} (\cdots\\
&\cdots \underset{\mathcal{K}_{6,5}}{\bowtie} (A(6,6,5) \underset{\mathcal{L}_{6,6}}{\bowtie} B(6,6,0) \underset{\mathcal{L}_{6,6}}{\bowtie} C(6,6,0)) \underset{\mathcal{K}_{6,6}}{\bowtie} \cdots\\
&\cdots) \underset{\mathcal{K}_{9,10}}{\bowtie} (\\
&(NA(10,1) \underset{\mathcal{L}_{10,1}}{\bowtie} NB(10,1) \underset{\mathcal{L}_{10,1}}{\bowtie} NC(10,1)) \underset{\mathcal{K}_{10,1}}{\bowtie}\\
&\dots \underset{\mathcal{K}_{10,9}}{\bowtie} (NA(10,10) \underset{\mathcal{L}_{10,10}}{\bowtie} NB(10,10) \underset{\mathcal{L}_{10,10}}{\bowtie} NC(10,10)))\\
&\underset{\mathcal{H}}{\bowtie}\\
&(E(1,1) \underset{\mathcal{N}_{1,1}}{\bowtie} \cdots \underset{\mathcal{N}_{1,9}}{\bowtie} E(1,10)) \underset{\mathcal{N}_{1,10}}{\bowtie} \cdots\\
&\cdots (E(6,1) \underset{\mathcal{N}_{6,1}}{\bowtie} \cdots Tm(6,6) \cdots \underset{\mathcal{N}_{6,9}}{\bowtie} E(6,10)) \cdots\\
&\cdots (E(10,1) \underset{\mathcal{N}_{10,1}}{\bowtie} \cdots \underset{\mathcal{N}_{10,9}}{\bowtie} E(10,10))
\end{aligned}
$$

$\mathcal{L}_{1,1} = \{r1(1,1), \mathit{bioon}(1,1), \mathit{biooff}(1,1)\}$
$\mathcal{K}_{1,1} = \{r5(1,1,1,2), r5(1,2,1,1)\}$
$\mathcal{K}_{1,9} = \{r5(1,9,1,10), r5(1,10,1,9)\}$
$\mathcal{L}_{1,10} = \{r1(1,10), \mathit{bioon}(1,10), \mathit{biooff}(1,10)\}$
$\mathcal{K}_{1,10} = \{r5(1,1,2,1),\ r5(2,1,1,1),\ r5(1,2,2,2),\ r5(2,2,1,2),\ r5(1,3,2,3),$

$r5(2,3,1,3),\ r5(1,4,2,4),\ r5(2,4,1,4),\ r5(1,5,2,5),\ r5(2,5,1,5),\ r5(1,6,2,6),$
$r5(2,6,1,6),\ r5(1,7,2,7),\ r5(2,7,1,7),\ r5(1,8,2,8),\ r5(2,8,1,8),\ r5(1,9,2,9),$
$r5(2,9,1,9),\ r5(1,10,2,10),\ r5(2,10,1,10)\}$
$\mathcal{H} = \{bioon(1,1),\ biooff(1,1),\ deathon(1,1),\ deathoff(1,1),\ growthon(1,1),\ growthoff(1,1),$
$bioon(1,2),\ biooff(1,2),\ deathon(1,2),\cdots,\ deathoff(10,9),\ growthon(10,9),\ growthoff(10,9),$
$bioon(10,10),\ biooff(10,10),\ deathon(10,10),\ deathoff(10,10),\ growthon(10,10),$
$growthoff(10,10)\}$
$\mathcal{N}_{1,9} = \{growth(1,9,1,10), growth(1,10,1,9)\}$
$\mathcal{N}_{1,10} = \{growth(1,1,2,1),\ growth(2,1,1,1),\ growth(1,2,2,2),\ growth(2,2,1,2),$
$growth(1,3,2,3),\ growth(2,3,1,3),\ growth(1,4,2,4),\ growth(2,4,1,4),$
$growth(1,5,2,5),\ growth(2,5,1,5),\ growth(1,6,2,6),\ growth(2,6,1,6),$
$growth(1,7,2,7),\ growth(2,7,1,7),\ growth(1,8,2,8),\ growth(2,8,1,8),$
$growth(1,9,2,9),\ growth(2,9,1,9),\ growth(1,10,2,10),\ growth(2,10,1,10)\}$

References

U. Alon. *An Introduction to Systems Biology*. Chapman & Hall/CRC, July 2006. 154

C. Athale, Y. Mansury, and T. S. Deisboeck. Simulating the impact of a molecular 'decision-process' on cellular phenotype and multicellular patterns in brain tumors. *Journal of Theoretical Biology*, 233:469–481, 2005. 26

J. C. M. Baeten, J. A. Bergstra, and J. W. Klop. Syntax and defining equations for an interrupt mechanism in process algebra. *Fundamenta Informaticae*, IX: 127–168, 1986. 31

N. Barkai and S. Leibler. Biological rhythms: Circadian clocks limited by noise. *Nature*, 403:267–268, 2000. 19

H. C. Berg. *Random Walks in Biology*. Princeton University Press, 1993. 24

M. Bernardo. Extended Markovian Process Algebra. In *Lecture Notes in Computer Science*, volume 1119, pages 315–330, 1996. 58, 81, 140

M. L. Blinov, J. R. Faeder, B. Goldstein, and W. S. Hlavacek. BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics*, 20(17):3289–3291, 2004. 2, 35

L. Bortolussi. Stochastic concurrent constraint programming. In *Electronic Notes in Theoretical Computer Science*, volume 164, pages 65–80, 2006. 2

G. E Briggs and J. B. S. Haldane. A note on the kinetics of enzyme action. *Biochem. J.*, 19:338–339, 1925. 14

M. Calder and J. Hillston. Process algebra modelling styles for biomolecular processes. In *Lecture Notes in Computer Science*, volume 5750, pages 1–25, 2009. 4, 34

M. Calder, S. Gilmore, and J. Hillston. Modelling the influence of RKIP on the ERK signalling pathway using the stochastic process algebra PEPA. In *Lecture Notes in Computer Science*, volume 4230, pages 1–23, 2006a. 3

M. Calder, V. Vyshemirsky, D. Gilbert, and R. Orton. Analysis of signalling pathways using continuous time Markov chains. *Transactions on Computational Systems Biology VI*, 4220:44–67, 2006b. 21, 40

L. Calzone, F. Fages, and S. Soliman. Biocham: An environment for modelling biological systems and formalizing experimental knowledge. *Bioinformatics*, 22:1805–1807, 2006. 2, 35

L. Cardelli. Brane calculi. In *Lecture Notes in Bioinformatics*, volume 3082, pages 257–278, 2005. 3, 34

L. Cardelli and A. Gordon. Mobile ambients. *Theoretical Computer Science*, 240 (1):177–213, 2000. 34

F. Ciocchetta and M. L. Guerriero. Modelling Biological Compartments in Bio-PEPA. In *Electronic Notes in Theoretical Computer Science*, volume 227, pages 77–95, 2009. 35

F. Ciocchetta and J. Hillston. Bio-PEPA: A framework for the modelling and analysis of biological systems. *Theoretical Computer Science*, 410:3065–3084, 2009. 2, 3, 33

F. Ciocchetta, A. Degasperi, J. Hillston, and M. Calder. Some investigations concerning the CTMC and the ODE model derived from Bio-PEPA. In *Electronic Notes in Theoretical Computer Science*, volume 229, pages 145–163, 2009. 22

R. Cleaveland and M. Hennessy. Priorities in process algebras. *Information and Computation*, 87(1-2):58–77, 1990. ISSN 0890-5401. doi: DOI:10.1016/0890-5401(90)90059-Q. URL http://www.sciencedirect.com/science/article/pii/089054019090059Q. Special Issue: Selections from 1988 IEEE Symposium on Logic in Computer Science. 31

J. O. Dada and P. Mendes. Multi-scale modelling and simulation in systems biology. *Integrative Biology*, 3:86–96, 2011. 3, 26

V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling of cellular signalling. In *Lecture Notes in Computer Science*, pages 17–41, 2007. 2, 35

A. Degasperi and M. Calder. Process algebra with hooks for models of pattern formation. In *CS2Bio 2010, Electronic Notes in Theoretical Computer Science*, volume 268, pages 31–47, 2010. iv, 92

A. Degasperi and M. Calder. Multi-scale modelling of biological systems in process algebra with multi-way synchronisation. In *CMSB 2011, to appear in ACM Digital Library*, 2011. iv

L. Dematté, C. Priami, and A. Romanel. The beta workbench: a computational tool to study the dynamics of biological systems. *Briefings in Bioinformatics*, 9(5):437–449, 2008. 35

Adam Duguid. An overview of the Bio-PEPA Eclipse plug-in. In *PASTA Workshop 2009*, 2009. URL `http://www.dcs.ed.ac.uk/home/stg/software/biopepa/`. 35

J. Elf and M. Ehrenberg. Spontaneous separation of bi-stable biochemical systems into spatial domains of opposite phases. *Systems Biology*, 1(2):230–236, 2004. 174

M.B. Elowitz and S. Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403:335–338, 2000. 19

R. Erban, J. Chapman, and P. Maini. A practical guide to stochastic simulations of reaction-diffusion processes. *ArXiv e-prints*, pages 1–35, 2007. 24

G. B. Ermentrout and L. Edelstein Keshet. Cellular Automata Approaches to Biological Modelling. *Journal of Theoretical Biology*, 160:97–133, 1993. 1, 25

M. A. Gibson and J. Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *The Journal of Physical Chemistry A*, 104(9):1876–1889, 2000. doi: 10.1021/jp993732q. URL `http://pubs.acs.org/doi/abs/10.1021/jp993732q`. 174

D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977. 16, 17

C. M. Guldberg and P. Waage. Über die chemische affinität. *J. Prakt. Chem.*, 19:69, 1879. 13

J. Heath, M. Kwiatkowska, G. Norman, D. Parker, and O. Tymchyshyn. Probabilistic model checking of complex biological pathways. *Theoretical Computer Science*, 391(3):239–257, 2008. ISSN 0304-3975. doi: 10.1016/j.tcs.2007.11.013. URL http://www.sciencedirect.com/science/article/pii/S0304397507008572. ¡ce:title¿Converging Sciences: Informatics and Biology¡/ce:title¿. 35

J. Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996. 2, 30, 33, 36, 37, 140

C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, Inc., 1985. 2, 30

M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, and H. Kitano. *Bioinformatics*, 19(4):524–531, 2003. doi: 10.1093/bioinformatics/btg015. URL http://bioinformatics.oxfordjournals.org/content/19/4/524.abstract. 1, 35

D. S. Jones and B. D. Sleeman. *Differential Equations and Mathematical Biology*. George Allen & Unwin Ltd, 1983. 24

J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. Van Nostrand, New York, 1960. 37

H. Kitano. Systems Biology: A Brief Overview. *Science*, 295(5560):1662–1664, 2002. 1

Hiroaki Kitano. A graphical notation for biochemical networks. *BIOSILICO*, 1(5):169–176, 2003. ISSN 1478-5382. doi: DOI:10.1016/S1478-5382(03)02380-1. URL http://www.sciencedirect.com/science/article/pii/S1478538203023801. 1, 35

E. Klipp, R. Herwig, A. Kowald, C. Wierling, and H. Lehrach. *Systems Biology in practice*. Wiley-Vch, 2005. 1, 9

J. Krivine, R. Milner, and A. Troina. Stochastic bigraphs. In *Electronic Notes in Theoretical Computer Science*, volume 218, pages 73–96, 2008. 35

T.G. Kurtz. The relationships between stochastic and deterministic models for chemical reactions. *The Journal of Chemical Physics*, 57(7):2976–2978, 1971. 21, 22

C. Laneve and F. Tarissan. A simple calculus for proteins and cells. In *Electronic Notes in Theoretical Computer Science*, volume 171, pages 139–154, 2007. 35

K. G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 91(1):1–28, 1991. 37

C. M. Macal and M. J. North. Tutorial on agent-based modelling and simulation. *Journal of Simulation*, 4:151–162, 2010. 25

H.H. McAdams and A. Arkin. It is a noisy business! genetic regulation at the nanomolar scale. *Trends in Genetics*, 15(2):65–69, 1999. 18

D. A. McQuarrie. Stochastic approach to chemical kinetics. *J. Appl. Prob.*, 4(3): 413–478, 1967. 15

H. Meinhardt. Models of biological pattern formation: from elementary steps to the organization of embryonic axes. *Curr. Top. Dev. Biol.*, 81:1–63, 2008. 1

R. Milner. *Communication and Concurrency*. Prentice-Hall, Inc., 1989. 2, 30, 31, 36

R. Milner. *Communicating and mobile systems: the π-calculus*. Cambridge University Press, 1999. 31

D. L. Nelson and M. M. Cox. *Lehninger Principles of Biochemistry, Fourth Edition*. W. H. Freeman, 2004. 9

David M. Nicol. Efficient simulation of internet worms. *ACM Trans. Model. Comput. Simul.*, 18:5:1–5:32, April 2008. ISSN 1049-3301. doi: http://doi.acm.org/10.1145/1346325.1346326. URL http://doi.acm.org/10.1145/1346325.1346326. 26

D. Noble. *The Music of Life: Biology Beyond the Genome*. Oxford University Press, 2006. 3, 92

N. H. Packard and S. Wolfram. Two-Dimensional Cellular Automata. *Journal of Statistical Physics*, 38:901–946, 1985. 25

M. Pedersen and G. Plotkin. A language for biochemical systems. In *Lecture Notes in Bioinformatics*, volume 5307, pages 63–82, 2008. 2, 35

A. Phillips and L. Cardelli. A Correct Abstract Machine for the Stochastic Pi-calculus. In *Concurrent Models in Molecular Biology*, 2004. 35

G. D. Plotkin. *A Structural Approach to Operational Semantics*. Lecture Notes, Aarhus University, 1981. 39

C. Priami and P. Quaglia. Beta-binders for biological interactions. In *Lecture Notes in Computer Science*, volume 3082, pages 20–33, 2005. 2, 3, 32

C. Priami, A. Regev, E. Shapiro, and W. Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Inf. Process. Lett.*, 80(1):25–31, 2001. 3

G. Păun and G. Rozenberg. A guide to membrane computing. *Theoretical Computer Science*, 287:73–100, 2002. 34

A. Regev. *Computational Systems Biology: A Calculus for Biomolecular Knowledge*. PhD thesis, Tel Aviv University, 2002. 42

A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical processes using the pi-calculus process algebra. In *Pacific Symposium on Biocomputing*, pages 459–470, 2001. 2, 31

A. Regev, E. Panina, W. Silverman, L. Cardelli, and E. Shapiro. Bioambients: An abstraction for biological compartments. *Theoretical Computer Science*, 325(1):141–167, 2004. 3, 34

S. M. Ross. *Stochastic Processes*. John Wiley & Sons, Inc., 1983. 19

I. H. Segel. *Enzyme Kinetics: Behaviour and Analysis of Rapid Equilibrium and Steady-State Enzyme Systems*. Wiley-Interscience, New York, 1993. 4, 13

C. Talcott. Pathway logic. In *SFM 2008, LNCS*, volume 5016, pages 21–53, 2008. 35

S. Tini. Non-expansive epsilon-bisimulations for probabilistic processes. *Theoretical Computer Science*, 411(22-24):2202–2222, 2010. 174

M. Tribastone, A. Duguid, and S. Gilmore. The pepa eclipse plug-in. *Performance Evaluation Review*, 36(4):28–33, 2009. URL http://www.dcs.ed.ac.uk/pepa/tools/. 35

D. C. Walker and J. Southgate. The virtual cell–a candidate co-ordinator for 'middle-out' modelling of biological systems. *Briefings in Bioinformatics*, 10 (4):450–461, 2009. x, 3, 26

D. C. Walker, N. T. Georgopoulos, and J. Southgate. From pathway to population - a multiscale model of juxtacrine egfr-mapk signalling. *BMC Systems Biology*, 2(1):102, 2008. 3

Z. Wang, L. Zhang, J. Sagotsky, and T. S. Deisboeck. Simulating non-small cell lung cancer with a multiscale agent-based model. *Journal of Theoretical Biology*, 233:469–481, 2005. 26

O. Wolkenhauer, M. Ullah, W. Kolch, and K. H. Cho. Modelling and simulation of intracellular dynamics: Choosing an appropriate framework. *IEEE Transactions on NanoBioScience*, 3:200–207, 2004. 17

L. Wolpert. The French flag problem: A Contribution to the Discussion on Pattern Formation and Regulation. *Towards a Theoretical Biology*, 1:125–133, 1968. 153

Index