A Process-Algebraic Approach to Modelling Multiagent Interaction Dynamics in Biological Systems

Thomas Wright

Master of Science by Research
Laboratory for Foundations of Computer Science
School of Informatics
University of Edinburgh
2017

Abstract

This project will develop new mathematical frameworks for modelling biological systems at the molecular, cellular, and ecosystem level by combining process algebra models that describe the behaviour of agents with multiagent interaction dynamics based on general kinetic laws. We will build upon Kwiatkowski and Stark's continuous π-calculus, a process algebra for modelling quantitative biochemical systems based on affinity networks, which complies to differential equations. To do this we will introduce three new process calculi: the mπI-calculus, a minimal extension of Sangiorgi's internal π-calculus to handle multiway interactions; the bond-calculus, a fully featured qualitative calculus; and the continuous bond-calculus, for modelling quantitative biological systems with general kinetic laws. Along the way we will extend affinity networks to cover multiway interactions with general kinetic laws, extend the familiar names and conames of the π-calculus to the new concepts of sites and locations, develop a new quantitative vector field semantics, and provide general encodings of autonomous differential equations and chemical reaction networks. Finally, we will apply our framework a range of real biological models: the Ping-Pong enzyme reaction mechanism, Lotka-Volterra Predator-Prey models, Kuznetsov's model of tumour immune interactions, and Elowitz and Stanislas' Repressilator.

Acknowledgements

I would like to thank my supervisors Ian Stark, Paul Jackson, and Boris Grot for their support and feedback throughout the year. I would particularly like to thank my primary supervisor, Ian Stark, for providing many interesting discussions and much invaluable advice throughout the year.

Next I would like to thank Juliet Cooke for reading drafts of this project, and for her constant support.

I would also like to thank the PEPA club and everyone else who attended my talks and discussed the ideas which make up this document, including Chris Banks, Simon Fowler, James McKinna, Jane Hillston, Simon Gilmore, Vashti Galpin, Ludovica Vissat, Anastasis Georgoulas, Paul Piho, Natalia Zoń, and Sam Lindley.

Thanks also to the once inhabitants of Office 1.07 for many great moments we have shared. In particular, I would also like to thank Wen Kokke for many interesting discussions and for introducing me to the many uses of cake in explaining process algebra, thus inspiring Example 3.2.8 (The Cake Pact).

This work was supported in part by the EPSRC Centre for Doctoral Training in Pervasive Parallelism, funded by the UK Engineering and Physical Sciences Research Council (grant EP/L01503X/1) and the University of Edinburgh.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Thomas Wright*)

Table of Contents

Chapter 1

Introduction

Over the last 50 years, the availability of ever more powerful computers and an influx of ideas from other sciences, has lead to a huge growth in the use of mathematical and computational models in biology. Frameworks such as differential equations and chemical reaction networks have increasingly been used to model the behaviour of complex biological systems, from chemistry, through cells, to ecosystems. The precise mathematical nature of these models finally allows us to start building a theoretical foundation for biology, with, for example, dynamical systems theory starting to shed light on the design principles underlying many biological mechanisms, whilst computers make it possible to perform detailed simulations, replacing much lab work with computational experiments, and allow us to go where no experiment can, including to the origins of life, and to life on other planets.

Mathematical biology is, however, arguable still in its infancy, with many models such as differential equations focusing on understanding the dynamics of individual components of biological systems in isolation, without shedding light on how the behaviour may be influence by their wider environment when embedded in real living systems. At the other extreme, systems biology tries to understand more complex biological systems in terms of their overall organisation, using methods such as chemical reaction networks to provide the circuit diagrams guiding the behaviour of biological systems. Whilst this has produced many successes, it is easy to get lost in the huge networks of interactions guiding even the simplest organisms; modern computers have already became far too complex to understand at the level of circuit diagrams, and biological systems are more complex still.

Biological process algebra takes a different approach, seeking to build new modelling languages, which describe biological systems in a modular and extensible manner as

concurrent systems with interacting components. This promises a number of major advantages over more established biological modelling techniques:

- Systems can be modelled in a compositional manner, by specifying the behaviour of each of their constituent agents, with new behaviour and new composite agents arising dynamically.
- Process algebras make it possible to give an abstract model of a system, collapsing complex components of a system into a single component of the model which focuses on its interface with the rest of the model components.
- As a formal language, a process algebra gives a precise description of a system, with a mathematical semantics giving meaning to models. A single model can then lead to many different mathematical analyses including logic and model checking, deterministic continuous state simulation as differential equations, or stochastic simulation as Continuous Time Markov Chains (CTMCs).

Process algebra has a long history in computer science, with process algebras such as Robin Milner's π-calculus [87] being developed to provide formal models of concurrent systems, in response to the new challenges posed by the rise of concurrency and parallel computing. Concurrency and parallelism are also key features of biological systems, and a wide variety of process algebras have been developed to model aspects of biological systems. Among these was Kwiatkowski and Stark's continuous π-calculus [80], which adapts the π-calculus to be better suited for biochemical modelling with new features such as a quantitative continuous state semantics based on differential equations, and affinity networks which capture reaction rates based on a network of compatible sites.

The continuous π-calculus has been successfully applied to modelling a number of biological systems including biological circuits and circadian clocks, but is limited in the range of biological systems it can model due to two key restrictions:

1. All interactions between agents must be modelled as binary interactions between pairs of agents.
2. The dynamics of all reactions must be described via the law of mass action, which states that the reaction rate is directly proportional to the concentration of the reactants.

The main aim of this dissertation will be to remove these restrictions, and extend the syntax and semantics of continuous-π to produce a new process algebra, the continuous

bond-calculus, which addresses these limitations. As we will see, removing these restrictions will significantly increase the range of biological systems we can model, allowing us to capture all chemical reaction networks, and, for the first time, consider models from a whole range of new areas including ecology, immunology, and gene regulation.

To do this we will first investigate qualitative calculi to define a sensible basis for modelling multiway interactions. Along the way, we will introduce *symmetrical multiway coordination*, a multiway synchronisation primitive with much of the power of the standard π-calculus send and receive operation. This will allow us to define the mπI-calculus, a minimal calculus for multiway interactions. Then we will introduce compatibility networks and sites and locations, a new syntax which simplifies the description of dynamic bonding of agents, and provides a clean separation between agent behaviour and site compatibility, allowing us to define a full qualitative language for multiway interactions in biological systems, the bond-calculus. Next we turn to the quantitative aspects of the language, developing a new vector field semantics for the language with full support for general kinetic laws. Finally, we will demonstrate that our new language is capable of capturing all chemical reaction networks, and moreover, any autonomous differential equation, and look at several modelling case studies to demonstrate its effectiveness in describing a wide range of biological systems.

Contributions

The main contributions of this dissertation are three new process calculi:

- The mπI-calculus – a minimal calculus for modelling multiway interactions, featuring a new communication primitive, *symmetric multiway coordination*.
- The bond-calculus – a qualitative calculus for biological modelling with multiway interactions. This features the new concepts of sites and locations, as well as compatibility networks, defining the compatibility between sites.
- The continuous bond-calculus – a quantitative extension of the bond-calculus for modelling continuous mixtures of species. This advances upon the continuous π-calculus by supporting multiway interactions and general kinetic laws.

For each of these calculi we have developed a formal operational semantics. In addition, the continuous bond-calculus has a new quantitative vector field semantics which allows quantitative simulation and derivation of differential equations, as well as an implementation of this semantics.

We also provide general encodings of autonomous differential equations and chemical reaction networks in the continuous bond-calculus to demonstrate its expressiveness.

Finally, we have developed adaptations of the following biological models as case studies for the continuous bond-calculus:

- The Ping-Pong mechanism for enzyme kinetics.
- A range of Lotka-Volterra Predator-Prey models.
- Kuznetsov's model of tumour immune interactions.
- Elowitz and Stanislas' Repressilator.

Dissertation structure

Chapter 1 is this introduction.

Chapter 2 introduces the wider context of the project and covers some technical background. It briefly introducing process algebra (including the π-calculus, and the πI-calculus), biochemical modelling with chemical reaction networks and differential equations, and the continuous π-calculus. It then surveys the literature on process algebra and other formal approaches to biological modelling.

Chapter 3 develops two qualitative calculi for biological modelling: the mπI-calculus and the bond-calculus. For mπI, we introduce the concept of *name parts*, and a new multiway communication primitive, *symmetric multiway coordination*, define the syntax of the language and an operation semantics, develop some examples, and finally prove a nice symmetry result. For the bond-calculus we introduce *compatibility networks* and develop the concept of *sites* and *locations*, before defining the syntax and semantics of the language, and returning to our examples to compare the two languages.

Chapter 4 extends the bond-calculus into the continuous bond-calculus, a quantitative calculus for biological modelling. First we will give a little background on qualitative calculi and general kinetics. Then we will see how affinity networks can be extended to handle general kinetics. Next we introduce a two level calculus for mixtures of species. Then we will define the formal syntax of the language, and the transition semantics for species. Next comes the process level semantics for mixtures which

defines the dynamics of the language. Finally, we discuss our implementation of the language.

Chapter 5 explores applications of the continuous bond-calculus in biological and chemical modelling. First we present general encoding results for autonomous differential equations and chemical reaction networks. Then we will present and evaluate our modelling case studies.

Chapter 6 is the conclusion. This will look back at the work in the project, before critically evaluating it in relation to related work, and suggesting future work.

Chapter 2

Background and Literature

This chapter will introduce some of the wider context of this project, and cover some technical background. We will first take a brief look at process algebras, and in particular, the π-calculus and πI-calculus. Then we will give a lightning introduction to biochemical modelling with differential equations and chemical reaction networks. Next we will given an introduction to the continuous π-calculus to see how it can be used for biological modelling, and explain the limitations which motivated this project. Finally, we will give a broader literature survey of other approaches to biochemical modelling using process algebras and other types of formal language.

2.1 Process algebra and the π-calculus

Many early computers operated in a sequential manner, with programs assumed to run sequentially, taking in input at the start, then running uninterrupted to perform computation on the input, and finally, producing a single output at the end [4]. Traditional models of computation follow this model also, with Turing machines [114] and the λ-calculus [29] both ultimately modelling computers as machines to calculate mathematical functions. However, the advent of time sharing, computer networking, parallel computing, object oriented programming, and graphical user interfaces soon strained this model, with modern computers spending at least as much time communicating and interacting, as computing functions. In the 1980s, process algebras such as Tony Hoare's CSP (Calculus of Sequential Processes) [71, 18], Bergstra and Klop's ACP (Algebra of Communicating Processes) [10], and Robin Milner's CCS (Calculus of Communicating Systems) [85] were introduced to give a richer theoretical model, using concurrency and synchronisation as the fundamental primitives of a theory of interacting systems. Later, Milner introduced

the π-calculus [88] which models mobile communication between processes using *channel names*: processes communicate by sending and receiving channel names, which makes it possible to represent dynamic networks of interacting processes, and gives a model rich enough to subsume both concurrent communication and sequential computation (in particular, to encode the λ-calculus [86]).

To give a basic example, in the π-calculus we can describe a system with *processes/agents* such as,

$$A(y) \triangleq a.\mathbf{0} + y(x).A(x)$$

which describes an agent $A(y)$ which can either *receives* (an empty message) on the channel *name* a to become $\mathbf{0}$ (the *null process*), or receives a new channel name x on the channel y and become $A(x)$. The behaviour of $A(y)$ will depend on the other processes which are available to communicate with it. For example, if we have a process,

$$B \triangleq \overline{a}.B$$

which uses the *coname* $\overline{a}$ of a, which represents sending on the channel a, then we can compose these two processes to form the parallel composition, $A(y) \mid B$. In this parallel composition, each process can either act independently, or they can communicate, when the receive operation represented by the name a with match up with the send operation represented by the coname $\overline{a}$ the parallel composition to evolve and become $\mathbf{0} \mid B \equiv B$. However, if we define a process,

$$C \triangleq \overline{y}\langle z\rangle.C$$

which sends the channel name z on the channel y to become C, then the parallel composition $A(y) \mid C$ can communicate to become $A(z) \mid C$.

Another key operation of the language is restriction, which makes it possible create new bound names private to a process. For example, if we define the process,

$$D \triangleq (\nu\, z)(\overline{y}\langle z\rangle.D'(z))$$

which creates a private channel name z and sends it on y, then the parallel composition $A(y) \mid D$ can communicate to become $(\nu\, z)(A(z) \mid D'(z))$ when $A(z)$ and $D'(z)$ now share the private channel name z, so only $D'(z)$ can communicate with $A(z)$ on z.

A key feature of process algebras is that they possess formal mathematical semantics which precisely characterise the behaviour of agents. For the π-calculus, this is given as

an operation semantics in the form of a *labelled transition system*, that is, set of *states* S, a set of *labels* Λ, and set of *labelled transitions* $\rightarrow \subseteq S \times \Lambda \times S$. So, for example, the process $A(y)$ has the labelled transitions,

$$A(y) \xrightarrow{a} \mathbf{0} \quad .$$
$$A(y) \xrightarrow[y(x)]{} A(x)$$

and, C has the transition system,

$$C \xrightarrow{\overline{y}\langle z \rangle} C$$

so their composition $A(y) \mid C$ has the transition system,

$$A(y) \mid C \xrightarrow{a} C$$
$$A(y) \mid C \xrightarrow{y(x)} A(x) \mid C$$
$$A(y) \mid C \xrightarrow{\tau} A(z) \mid C$$

which contains all of the transitions that both $A(z)$ and C can make individually, along with a new *silent transition* (marked with τ)

$$A(y) \mid C \xrightarrow{\tau} A(z) \mid C$$

arising from the communication between them.

The π-calculus' ability to send and receive channel names adds a number of complications to its semantics that simpler calculi such as CCS do not share:

- The states in the transition system need to be specified as equivalence classes of a structural congruence relation $\equiv$, which identifies syntactically equivalent processes.
- There is no single clear definition of semantically/behaviourally equivalent processes. Instead, there is a wide variety of different equivalences with different levels of granularity.
- There is a distinct asymmetry between the send and receive operations, as whilst the receive operation binds a new channel name, send does not.

One may ask how many of these complications are essential for a calculus of mobile processes? The answer is, in fact, none of them, excepting a congruence $\equiv_\alpha$ which identifies α-equivalent processes[1]. Davide Sangiorgi introduced the πI-calculus [105],

[1]Two processes are α-equivalent if they differ only in the names given to bound variables.

which replaces send and receive with a single communication operation, which allows two processes to coordinate to receive a fresh shared name. For example, this would still allow us to express A and D as,

$$A(y) \triangleq a.\mathbf{0} + y(x).A(x) \qquad\qquad D \triangleq \overline{y}(z).D'(z)$$

and communicate on the name y/coname $\overline{y}$ as before to produce the process $(\nu\, z)(A(x) \mid D'(z))$. This is equivalent to restricting the π-calculus such that all sends must be of a freshly bound name, so, for example, process D would be allowed, whilst process C is not. This simple restriction resolves all of the complications listed above, adding to CCS only the question of α-equivalence, whilst preserving much of the expressiveness of the π-calculus, including the ability to express data types and the λ-calculus.

In Section 2.4.1 we survey some other process algebras which have developed for biological modelling including many variants of the π-calculus. For the moment, we have paid particular attention to πI since it will form the basis of all the process calculi we will develop in later chapters.

2.2 Biochemical modelling 101

Suppose you find yourself saddled with the ambitious task of understanding life. After first taking a few moments to despair at the enormity of your undertaking, you decide if you are to understand life, you must first understand what it is made of. You journey down through organisms, organs, tissues, cells, until finally, at the bottom of it all you find a mess of chemicals. At this point you get excited, you have found the basic building blocks of life. So you proceed to examine every living thing you can think of, and continue to find more chemicals. You catalogue and organise these chemicals and their many intricate structures. You start to link certain type of chemicals, to different functions of organisms: you find a wide range of proteins doing most of the real work, whilst DNA carries the blueprints they follow. At this point you have a pretty good idea of what life is made of, but cannot shake the feeling that you are still a long away from understanding life. For the intricacies of life are not contained in the snapshot of an instant, however detailed, but in how the parts move and interact over time. Hence you must now attempt a second stage of your journey, to take your chemical building blocks of life, and begin to model their dynamics and interactions.

This endeavour can itself be split into parts. First you must identify all of the reactions and chemical processes in taking place in a living being and how they fit together. Just

Figure 2.1: *An enzyme reaction.*

like in understanding computers, we must find a circuit diagram detailing how all of the components are connected and feed into each other; this circuit diagram is known as a *Chemical Reaction Network*. Next we must understand the rates of each of these reactions, and how these depend on their inputs, and the wider environment; that is, we must understand the *kinetic laws* underlying the reactions. The reaction kinetics is of utmost importance to understanding the behaviour of biochemical systems. Since life is made up of analogue systems, with no centralized clocks[2], a small difference in reaction rates can radically alter the qualitative behaviour of the system. For both of these tasks, computer modelling and simulation can play a key role in storing the large networks of reactions, and simulating their kinetics.

Since we have to start somewhere, we will start by looking at the example of enzymatic reactions – one of the most prevalent patterns of reaction throughout life. If we have a reaction in which a substrate S is broken down into a product P, but only in the presence of an enzyme E, we can represent this as a scheme of reactions,

$$S \quad + \quad E \quad \rightleftharpoons \quad C \quad \longrightarrow \quad P \quad + \quad E.$$

In this reaction the substrate S and the enzyme E first bind together to form a complex C which then produces a product P and releases the enzyme E. The first stage of the reaction is a reversible reaction $S + E \rightleftharpoons C$ in which the complex C is being both formed and broken at the same time.

In order to understand the dynamic behaviour of these reactions, we need to describe how the rate of the reaction depends on the concentration of the reactions – the kinetic law governing the reaction. For basic chemical reactions, the kinetics is described by the *law of mass action*, which states that the rate is determined as the product of the concentration of the reactants, multiplied by a constant, the *stoichiometric rate* of the reaction. This allows us to rewrite our enzyme reaction, this time with the stoichiometric

[2]In many organisms, many reactions are coupled to oscillators, but in these cases the exact kinetics becomes all the more critical.

rates included,

$$S \quad + \quad E \quad \underset{r_{-1}}{\overset{r_1}{\rightleftharpoons}} \quad C \quad \xrightarrow{r_2} \quad P \quad + \quad E.$$

This reaction network completely captures the dynamics of our enzyme system, but we need to be more explicit about what all of this means in order to simulate and analyse the dynamics. The most common way to express these dynamics is as a system of differential equations; for example, the reactions in our enzyme example correspond to the equations,

$$\begin{aligned}
\frac{\mathrm{d}\,[S]}{\mathrm{d}t} &= -r_1\,[S]\,[E] + r_{-1}\,[C] \\
\frac{\mathrm{d}\,[E]}{\mathrm{d}t} &= -r_1\,[S]\,[E] + r_{-1}\,[C] + r_2\,[C] \\
\frac{\mathrm{d}\,[C]}{\mathrm{d}t} &= r_1\,[S]\,[E] - r_{-1}\,[C] - r_2\,[C] \\
\frac{\mathrm{d}\,[P]}{\mathrm{d}t} &= r_2\,[C]\,,
\end{aligned}$$

where $[S]$, $[E]$, $[C]$, and $[P]$ represent the concentrations of S, E, C, and P respectively. This corresponds directly to the information from the reaction diagram, using the law of mass action to derive the reaction rates. So, for example, in the equation for $\frac{\mathrm{d}[C]}{\mathrm{d}t}$ the term $r_1\,[S]\,[E]$ comes as S and E are reacting to form C at stoichiometric rate r_1, whilst the term $-r_{-1}\,[C]$ describes the reverse reaction breaking C down at stoichiometric rate r_{-1}, and the term $r_2\,[C]$ is releasing a product P and the enzyme E at stoichiometric rate r_2.

This framework of modelling systems as a network of reactions, coupled with reaction rates, is one of the most widespread approaches in mathematical modelling of biochemical systems. It also spreads far beyond chemical reactions, with systems such as gene regulation, competition between species in ecology, and interactions between pathogens and immune cells also being treated as networks of chemical reactions. However, whilst this approach is effective at capturing the behaviour of our system in isolation, in biochemical systems very little happens in isolation, and the behaviour of a system may change drastically when combined with other reactants. Both our reaction based view, and the system of differential equations tell us nothing about how our system combines with other components of a larger system. For example, it is possible to block the action of our enzyme by an *inhibitor* I, which binds to the enzyme in competition with the substrate, decreasing the overall rate of the reaction. In this case, we add a reaction, $E + I \underset{r_{-3}}{\overset{r_3}{\rightleftharpoons}} EI$, describing the interactions between enzyme E and our inhibitor I. Just when we thought we were done describing the behaviour of our enzyme, we need

to add more rules every time we put it together with a new chemical. Things get even worse when we look at the differential equations, as we have to revise each of our existing equations to account for reactions with the new chemical. In order to overcome these limitations, we need a way to build *compositional* models of biological systems, which may be combined as building blocks of larger models, or embedded within larger systems representing their environment.

2.3 Biochemical modelling in the continuous π-calculus

The problem of non-compositionality of models we encounter in conventional methods of biological modelling mirrors the limitations of modelling computer programs as isolated sequential processes, motivating us to pursue the same solution, using process algebras to provide compositional models of biological systems. Whilst both biological systems and computer systems consist of many agents interacting in parallel, many classical models of computer systems are discrete or non-deterministic, whereas biological systems are analogue systems, with transitions occurring at quantitative rates, and little central synchronisation. This motivated the development of new process algebras to capture the quantitative nature of biological systems. Amongst these is Kwiatkowski and Stark's continuous π-calculus [80], which extends the π-calculus with *affinity networks*, which replace names and conames with a network of *sites*, with edges between them labelled by real valued affinities. We then let two sites engage in a reaction when they are joined by an edge in the affinity network, with the reaction rate determined by the affinity according to the law of mass action.

2.3.1 Sites and affinity networks

We can better understand the concepts of sites and affinities by returning to our enzyme example and considering why reactions happen. The enzyme E is able to bind to the substrate S since their chemical structures are compatible. At the lowest levels, this can be compared by comparing the chemical structure of E and S, however we are able to abstract this by viewing molecules as composed of *reaction sites* which may bind to other *sites* depending on the degree of affinity between them. For example, if our enzyme E possesses a reaction site e and the substrate S possesses a site s, then the reaction between them is explained by the degree of compatibility between these sites (see Figure 2.2).

This site centred view is at the core of modelling in continuous π. We specify the

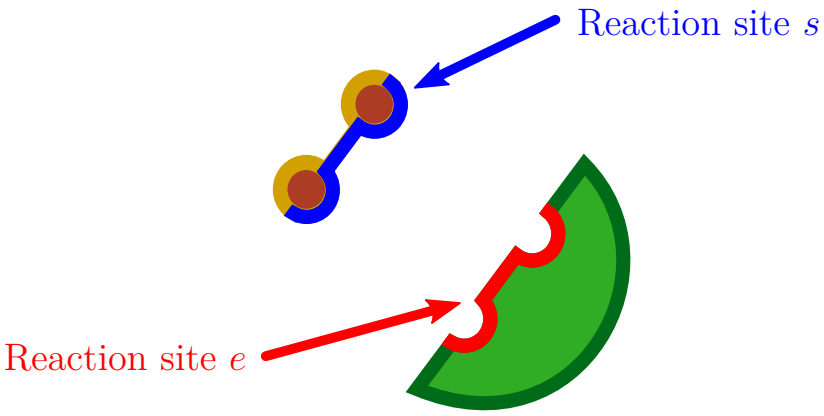


Figure 2.2: *An illustration of the substrate S and enzyme E, and their respective reaction sites s and e.*

behaviour of all of our agents by listing their sites, and how their structure evolves after they become bound. In continuous π, we also allow agents to send and receive site name when binding on a site, so they may have internal sites, which change the behaviour of the agent to which they have became bound. So we can define our enzyme as,

$$E \triangleq e(x).(x.E)$$

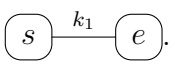
Breaking this down, this means that E can react on a site e. Upon reaction, E will receive a new site x with which it can react with the species to which it has just bound. After binding E will evolve into the species $x.E$, meaning that it can now react at site x to be released. The other half of the coin is the substrate S which we can define as,

$$S \triangleq \left(\nu \; r \overset{k_{-1}}{\text{---}} x \overset{k_2}{\text{---}} p\right) s\langle x\rangle.(r.S + p.P).$$

This the first part of the definition is a *local affinity network*, which says the internal site r and x react together at rate r, and the internal sites x and p react together at rate k_p. Next we have the output prefix $s\langle x\rangle$ which says that S can bind at site s, giving access to its internal site x to the agent it has bound with. Then it evolves into $r.S + p.P$, which means it can either react on site r and release S, or react on site p and release the product P. Finally we are in a position to model our complete system of reactions as the process,

$$\Pi = [E] \cdot E \parallel [S] \cdot S \parallel [P] \cdot P,$$

and define the interactions between the external sites e and s via the global affinity network,



The agent based nature of this model means that continuous π is able to infer the behaviour of the whole system from our descriptions of each component. For example, given the two components E and S, we can infer they can react at rate $r_1 [E] [S]$ from their sites and the affinity network $\mathcal{M}$, and using the same rules derived for sharing names between concurrent processes in Milner's π-calculus, we can deduce that they combine to form a new species,

$$C \triangleq \left(\nu \; \boxed{r} \overset{k_{-1}}{\text{———}} \boxed{x} \overset{k_2}{\text{———}} \boxed{p}\right) \Big((x.E)|(r.S + p.P)\Big).$$

The benefit of the compositionality we have gained in this modelling approach becomes apparent when we attempt to add a new species, an inhibitor I. The inhibitor I has a site i which binds competitively to the enzymatic site e, to form another complex EI, and so reduce the rate of the reaction by reducing the amount of usable enzyme. In continuous π, we simply need to define a new species for the inhibitor,

$$I = \left(\nu \; \boxed{u} \overset{k_{-3}}{\text{———}} \boxed{x}\right) i\langle x\rangle.r.I.$$

This species can bind on site i, and pass a name x which gives access to its internal site r. Reacting on r will then cause it to unbind, releasing I once more. Now, we just need to add the new site interaction $\boxed{i} \overset{k_3}{\text{———}} \boxed{e}$ to the global affinity network, and our model will take into account the added effects of the inhibitor.

2.3.2 Continuous semantics

Continuous π uses a continuous vector field semantics in order to allow quantitative simulation of models, by extracting systems of differential equations. This makes it possible to efficiently simulate systems regardless of the number of copies of each species of agent involved in the system, however, it ignores all stochastic effects of the system, and hence may not be accurate in systems where stochasticity plays a significant role, or where the population is not sufficiently high to for a fluid/continuous approximation to make sense.

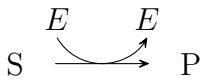
The semantics represents each process Π by two vectors, the *immediate behaviour* $\frac{\mathrm{d}\Pi}{\mathrm{d}t}$ which represents the velocity of instantaneous evolution starting at the process, and the

potential behaviour $\partial\Pi$ which contains information about the potential transitions the process may make [80]. One key feature of the semantics, is it is compositional, allowing the semantics of a process to be built up from from the semantics of a constituent parts.

2.3.3 The limitations of binary mass action reactions

The design and semantics of continuous π is based on the fundamental assumption that every reaction is a binary reaction governed by the law of mass action[3]. This should, in theory, be a sound assumption for chemical modelling since it has long been a central tenant of chemical kinetics, that any reaction, however complicated, can be reduced to a sequence of reactions of this type, known as the *mechanism* of the reaction. In practice this is easier said than done, as whilst the overall reaction rate can be readily observed phenomenologically, finding the mechanism is much harder, often requiring decades of painstaking experimental and theoretical work. Moreover, even if we do know the mechanism of a reaction, it is often possible to accurately describe the reaction with a single rate law, whilst the underling mass action reactions have many more rate parameters which would need to be fit to experimental data. Therefore, *general kinetic laws* and *non-binary reactions* are ubiquitous in real biochemical models, providing a major limitation to the range of models the continuous π-calculus is able to express[4].

We can see an example of this in our enzyme system, which would often be modelled as a single reaction,



with reaction rate ν governed by the Michaelis-Menten kinetic law,

$$\nu = \frac{\mathrm{d}\,[P]}{\mathrm{d}t} = -\frac{\mathrm{d}\,[S]}{\mathrm{d}t} = \frac{V_{\max}\,[S]\,[E]}{K + [S]},$$

where $V_{\max}$ is the maximum reaction rate (for saturating substrate concentration), and K is the substrate concentration at which the reaction rate reaches half of its maximum value. This gives a good approximation of the underlying mass action reactions under certain conditions, but it is frequently used more widely, as a general model of enzyme

[3] Or possibly a unary reaction, handled by a special prefix τ_r, denoting exponential decay at rate r

[4] Stanley Wang's dissertation [104] did show that it is actually possible to express some models based on general kinetic laws in continuous π by using the underlying mass actions models and fitting the parameters of the mass action model to match the general kinetic law, however, this produces significantly more complicated models, and there is no guarantee that the model parameters this produces will match biological reality.

kinetics with parameters fit to experimental data (and, hopefully, subject to experimental validation).

This limitation also restricts the applicability of the continuous π-calculus in areas outside biochemical modelling, such as ecological modelling, or immunological modelling, since here phenomenological rate laws with no basis in mass action kinetics are common.

2.4 Literature survey

The continuous π-calculus is just one amongst many process algebra which have been applied to biological modelling; in this section we give a brief tour of the area, focusing particularly on applications of process algebra to *biochemical* modelling, before looking more broadly at other paradigms for formally modelling biological systems.

2.4.1 Process algebra in biological modelling

The use of process algebra in modelling biochemistry started with Regev et al [103], who argued that the π-calculus was suitable for modelling biochemical pathways including transcriptional circuits, metabolic pathways, and signal transcription networks, and demonstrated the approach by modelling the RTK MAPK signal transduction pathway. Regev introduced the 'molecule as process' abstraction, mapping the molecules in a biochemical system to processes in a process algebra. Since then there has been a proliferation of different approaches to extend traditional process algebra to better model biochemical pathways. These include the CCS-R process algebra which was introduced to model biochemical systems with reversible reactions [43], and the causal/enhanced π-calculus which records the causal relationship between reactions in a pathway [40, 39]. Other custom process algebras were developed to model specific aspect of biological systems including Regev's BioAmbients [102] (which extends Cardelli and Gordon's Mobile Ambients [28]) which models compartmentalized biological systems (where reactions can be restricted to within ambient locations such as a cell membrane), Membrane Calculi such as the Brane Calculus [25], and the Bitonal/Atonal Calculus [26], and Strand Algebra [27] which models the DNA strand reactions which form the basic of DNA computing.

Whilst initial applications of process algebra to biochemical systems used non-deterministic process algebras and focused on the qualitative aspects of systems, real biological systems are analogue systems progressing at quantitative rates, and subject to stochastic noise. Markovian process algebras such as PEPA [70], use transition systems labelled with

quantitative transition rates (which are taken as the parameter of an exponential distribution), and have stochastic semantics based on Continuous Time Markov Chains (CTMCs). Whilst Markovian Process Algebras were originally developed for performance analysis of distributed computer systems, in [19] Calder, Gilmore, and Hillston applied PEPA to modelling the influence of RKIP on the ERK signalling pathway This work demonstrated both the ability of Markovian Process Algebra to capture the quantitative aspects of biochemical pathways, and the potential for more abstract representations of biological systems, with processes corresponding to not just individual molecules but more abstract entities such as subpathways. New stochastic process algebras were also developed for biological modelling including the Stochastic π-calculus [98] which supports simulation of models using on Gillespie's Stochastic Simulation Algorithm and the law of mass action, and Bio-PEPA [34] which is based on PEPA with a syntax more directly tailored to biochemical modelling and direct support for stoichiometry, multiway synchronisation, and general kinetics. Stochastic Concurrent Constraint Programming (sCCP) [13] is another stochastic process algebra with synchronisation based on constraints on a shared store, which has also been applied to biochemical modelling including general kinetics.

A major hurdle to stochastic simulation is the exponential increase size of the state space with the size of the system, making simulating large systems extremely computationally challenging. Hillston tackled this problem with the Fluid Flow Approximation [69] for PEPA which derives systems of differential equations from PEPA and Bio-PEPA models. Using a continuous state model allows for much more scalable simulation using standard numerical techniques, at the expense of ignoring stochastic fluctuations. In [65] it is shown that the differential equations derived from the model approximate the average limiting behaviour of the stochastic system with sufficiently high agent populations, however, as examples in [15], it is not guaranteed that the dynamic evolution of the systems match due to sensitivity to initial conditions. The continuous-π calculus [80] has a different continuous state semantics which attempts to derive differential equations in a more compositional way in terms of real vector spaces; this is however tied to mass action kinetics. Stefanek developed a continuous (and spatial) extension of the stochastic π-calculus [1], but also considered systems with infinitely many agents (corresponding to infinite systems of differential equations). There is also a translation from sCCP to differential equations [15, 14], and the inverse translation has also been considered. Another approach which attempts to allow more efficient simulation of systems, whilst still capturing some of the effects of stochastic noise, is Stochastic Differential Equations, which combine differential equations with a noise term, and have been applied to interpreting

PEPA models [109].

There are still some aspects of the quantitative dynamics of biological systems which are not captured in either the stochastic/Markovian or continuous approach which have lead to the development of other quantitative process algebras. One of these is the effect of latency/time delays, which can be modelled in Bio-PEPAd and simulated as Delay Differential Equations or Generalized Semi-Markov Processes [23]. Many biological systems also include discontinuous global transitions, making their qualitative behaviour impossible to capture a continuous setting, and difficult to approximate, as these correspond to stiff equations which are notoriously difficult to simulation. These systems have been tackled by hybrid process algebras such as Bio-PEPA with Events [31], sCCP [16, 17], and HYPE [62].

Whilst traditional process algebras do not consider the spatial aspects of systems treating agents as existing in a well-mixed solution, the agents of many biochemical systems are distributed across a more interesting spatial structure with, for example, cells containing many different compartments, separated by membranes. Whilst special purpose calculi such as BioAmbients [102] modelled these spatial features, β-binders [99] takes a different approach adding compartments to the π-calculus; in this approach membranes are represented as boxes with sites defining their interface, but actual agents are described as in the π-calculus. Other languages including Bio-PEPA with Compartments [32], PALPS [52], PALOMA [57], and MELA [117] have covered modelling ecological systems with agents at spatially distributed locations. All of these approaches required adding quite extensive extensions to the languages to handle distribution. Carbone and Matteis [24] showed that distribution can be captured with a minimal expansion to the π-calculus, the operation of polyadic synchronisation, which allows processes to synchronise atomically on a list of channel names. The π@-calculus applied polyadic synchronisation and priorities to biological modelling and showed this is sufficient to encode BioAmbients, the Brane Calculus, and β-binders [116, 20].

The minimal mathematical syntax characteristic of formal calculi is a barrier to broader adoption of process algebra in biology. There have been a number of attempts to build higher level languages onto of process algebra, to ease the process of biological modelling. A. Phillips introduced a graphical representation of processes in the stochastic π-calculus [94, 93]. BlenX builds on β-binders to a biological modelling languages incorporating the abstraction of affinity networks to describe the reactivity between sites, and of biological entities as boxes with sites [47]. Other work including SPiCO [78] and [51] linked process algebra modelling with concepts from object oriented program-

ming, recasting agents as multi-profile concurrent objects, to build more modular and understandable models. Other languages have moved process algebra modelling closer to more general concurrent programming, where agents may have attributes/variables and guard communication based on the values of these variables. This approach was introduced by the attributed π-calculus [74] which allows functional expressions on attributes to guard communication, and extended by the imperative π-calculus [73] which also allows communication to modify these attributes. This approach simplifies many models, is flexible enough to provide simple encodings spatial structure (including encoding $\pi@$) and SPiCO [74, 75], and also making it possible to describe numerical attributes of spatial compartments such as volume.

2.4.2 Other approaches to formal biological modelling

An early attempt to model biological systems in a formal calculus by Fontana [59] who used the λ-calculus to model molecules, with reactions simulated according to mass action kinetics, and explores questions of self-organisation and the origins of life. Whilst this was pioneering work, function application is not the most natural model for molecular binding and simulation of reactions occurring between molecules in solution had to be handled outside of the formal calculus, so other formal calculi have dominated subsequent research.

Another approach to biochemical modelling is rule based modelling, in which agents are represented using an abstract model of their structure, and the dynamics of the system are represented as rewriting rules, which match and rewrite given structural patterns. This approach is followed by Kappa [44] which models proteins as nodes on a graph with attached sites, and allows reactions when the interfaces of nodes matches the pattern specified in a reaction rule. Kappa has non-deterministic, continuous, and stochastic interpretations and has been given semantics in terms of both single pushout graph rewriting [42, 68], and a translation into the π-calculus [44]. BIOCHAM is a rule based language for modelling chemical pathways with special syntax to represent complexation and phosphorylation of sites [55]. It has support for analysing models as non-deterministic, continuous state, or stochastic systems. ℓ is another rule based approach to biochemical modelling [120]. It is implemented as an embedded domain specific language, compiling to stochastic simulation code in C#, and has support for general kinetics and dynamic complex formation. LBS attempts to give a modular rule based description of biological systems [91].

Milner's Bigraphical Rewriting Systems [72] were conceived as a unifying framework for modelling concurrent and mobile systems, in a framework based on precategories, and rewriting rules defining reactions, which subsume aspects of both mobile process calculi such as the π-calculus, and spatial calcului including mobile ambients. Various languages based on the bigraphs framework have been developed for biological modelling including Stochastic Bigraphs [77], the $\mathcal{C}$-calculus [41], Bio-β [6], and Biological Bigraphs [5].

Petri nets take a different approach, modelling biochemical pathways as networks and the dynamics as a flow of tokens through the network [92, 66]. They were originally used for non-deterministic modelling, but have stochastic, hybrid, and continuous interpretations [45]. It is also possible to model pathways as boolean networks, with steps represented as binary on/off choices [112]. This logical representation simplifies analysis and is enough to give quite a lot of insight into the dynamics of the system, but does not include the quantitative aspects of the system (although there are extensions of the framework in this direction). P systems [101] arose as a biologically inspired models of computing, which model computation at a membrane, and gave rise to the field of membrane computing. A probabilistic extension of P systems has been applied to biological modelling [3]. L systems are another early formal model of biological biological systems based on rewriting rules, and have been used to study plant growth [100].

Chapter 3

Qualitative Calculi for Multiway Interactions

In this chapter we will build two qualitative calculi for modelling multiway interactions: the mπI-calculus, a minimal calculus for multiway interactions, and the bond-calculus, a more extensive calculus featuring compatibility networks and sites and locations. First, we will give a little background to the problem of multiway synchronisation in process algebras and biochemical modelling. Then we will develop the mπI-calculus, introducing name parts, defining the syntax of the language, specifying by a transition semantics, and then giving a range of examples modelling multiway synchronisations in the language, before finally proving a nice symmetry property for name parts. Next we will develop the bond-calculus, first introducing compatibility networks and sites and locations, then defining the syntax and transition semantics for the language, and finally, revisiting our examples to compare the two languages.

3.1 Introduction

Historically, process algebra have taken two main approaches to synchronisation, with languages such as CSP and PEPA supporting multiway synchronisation between arbitrarily many agents on a list of names, whilst languages such as CCS and the π-calculus use binary communications based on send and receive primitives, which match names to conames. This limitation to binary synchronisation has not proven fatal to those languages in the second camp, since any n-way synchronisation can be implemented as a sequence of binary communications (real programs communicating across a network must also implement n-way synchronisation in this manner), and, in fact, this division between

send and receive is closely linked to the π-calculus' most distinctive feature, the mobile passing of channel names between agents. Hence, multiway synchronisation has remained an uncommon feature among name passing calculi.

Just as with processes, all chemical reactions can be reduce to binary interactions between molecules, however, the challenge of finding all of the intermediate binary interactions corresponding to a single observed chemical reaction (called the *mechanism* of the reaction), has provided one of the most difficult problems in chemistry, with the mechanism of many reactions still debated to this day. Moreover, once we consider modelling reactions quantitatively, we will see that reducing a reaction to a sequence of binary reactions introduces many more parameters to the model, each of which would require additional costly experiments to estimate. Therefore, in practice most biological models treat multiway reactions as a single step. This is reflected in formalisms such as chemical reaction networks, which model chemical reactions as transitions which atomically transform n reactants into m products.

The application of process algebra to biochemical modelling started with the π-calculus, which can represent molecules as processes, and reactions as communication. Given the nature of communications in the π-calculus, all reactions must be modelled as binary interactions between molecules (corresponding to spending and receiving names between agents), corresponding to specifying an explicit mechanism for the reaction. Despite this limitation the π-calculus has been successfully been applied to model a range of biochemical pathways, however, the restriction remains a major limitation for practical biological modelling. Several attempts have been made to extend variants of the π-calculus with transactions [35, 36, 30], which allow multiple binary interactions implementing an n-reactant reaction to be treated as a single atomic action, but whilst this does allow such reaction to be represented, it requires a rather extensive extension of the π-calculus with concepts foreign to biologists, and still requires the mechanism of the reaction to be specified.

The Bio-PEPA language takes a different approach, using CSP style multiway synchronisation to directly represent n-reactant reactions [33]. This proved very successful, allowing the a full range of reactions to expressed concisely and intuitively. The drawback, however, is that in restricting ourself to CSP style synchronisation, we lose the expressiveness of the π-calculus' dynamic name binding, and along with it, some of the most interesting features of chemical reactions, such as dynamic binding of existing molecules to form new molecules, and polymerisation, where arbitrarily long chains of molecules are formed from a finite set of reactants. Instead of the 'molecule as a process' abstraction,

the 'species as a process' abstraction is adopted, in which processes correspond to the chemical species involved in a reaction, and agents corresponding to each of the species involved in the reactions must be explicitly listed.

One may ask it it possible to have the best of both worlds, naturally representing arbitrary n-reactant reactions using multiway synchronisation as in Bio-PEPA, whilst retaining the expressiveness of π-calculus style name passing? In this section we will attempt to do just that, introducing a new communication primitive, multiway internal coordination, which overcomes the limitations of both CSP style multiway synchronisation, and π-calculus style binary name passing. By extending the notion of internal mobility (as introduced in Sangiorgi's internal π-calculus (πI) [105]) to n processes, this allows them to synchronise atomically in a symmetric manner, whilst providing them a shared channel for future coordination. We will see that this mechanism retains much of the power of name passing in the π-calculus, allowing us to represent dynamic complex formation, and polymerization.

To do this we will introduce two new process calculi. We will first consider the mπI-calculus, a minimal extension of the π-calculus to include multiway internal coordination, allowing us to discuss the implications of this communication mechanism. We will then move on to considering *compatibility networks*, which provide an intuitive and visual way of specifying the reactivity between sites in a biological context. This will lead us to consider how finite, static compatibility network can be reconciled with the infinite and dynamically changing multiway reactions we wish to model, and hence, to split the notion of channel names into sites and locations. Together these ideas lead us to the bond-calculus, a higher level and more intuitive language, combining compatibility networks, sites and locations, and multiway internal coordination.

3.2 mπI: A minimal calculus for multiway coordination

In this subsection we define the mπI-calculus, which extends concept of internal mobility of the πI calculus, to a more general synchronisation mechanism, to allow n-way binding between different processes. This works by generalising the familiar names and conames of the π-calculus, which allow a channel name to be cut into two halves, one half denoting the sending end of the channel, and the other half the receiving end, to a more general notion of *name parts*, which allows a name to broken into as many parts as one likes. So for example, the name x can be broken up into a list of n names parts, $x^{[1/n]}, x^{[2/n]}, \ldots, x^{[n/n]}$.

Definition 3.2.1 (Name parts)**.** Given a set of names $\mathcal{N}$ and a name $x \in \mathcal{N}$, the *part names* of x are $x^{[1/n]}, x^{[2/n]}, \ldots, x^{[n/n]}$. We then denote the set of all part names as $\mathcal{N}_{\text{part}}$.

In ordinary variants of the π-calculus, if you bring a name and a coname together they will react. In the mπI-calculus, bringing all of the part names of a name together will cause a reaction,

$$a^{[1/n]}(l).P_1 \mid a^{[2/n]}(l).P_2 \mid \ldots \mid a^{[n/n]}(l).P_n \xrightarrow{\tau} (\nu\, l)(P_1 \mid P_2 \mid \ldots \mid P_n) \qquad \text{(a)}$$

However, in order to enable these multiway reactions we want to build up a sequence of potential reactions, which include some, but not all of the required part names. These will denoted as transitions from processes P to abstractions $(l)(P_l)$, which contain a process P_l abstracted over an unknown name l. For example, we want to be able to build up partial reactions such as,

$$a^{[1/n]}(l).P_1 \mid a^{[3/n]}(l).P_3 \xrightarrow{\wr a^{[1/n]}, a^{[3/n]} \wr} (l)(P_1 \mid P_3)$$

and,

$$a^{[2/n]}(l).P_2 \mid a^{[5/n]}.P_5 \xrightarrow{\wr a^{[2/n]}, a^{[5/n]} \wr} (l)(P_2 \mid P_5)$$

into larger partial reactions such as,

$$a^{[1/n]}(l).P_1 \mid a^{[2/n]}(l).P \mid a^{[3/n]}(l).P_3 \mid a^{[5/n]}(l).P_5 \xrightarrow{\wr a^{[1/n]}, a^{[2/n]}, \cdot a^{[3/n]}, a^{[5/n]} \wr} (l)(P_1 \mid P_3 \mid P_2 \mid P_5)$$

When we have built up a potential reaction with all of the parts of a name such as,

$$a^{[1/n]}(l).P_1 \mid a^{[2/n]}(l).P_2 \mid \ldots \mid a^{[n/n]}(l).P_n \xrightarrow{\wr a^{[1/n]}, \ldots, a^{[n/n]} \wr} (l)(P_1 \mid P_2 \mid \ldots \mid P_n)$$

it will be able to be committed at any stage, forming the final reaction (a). The parts of a name can be brought together in any order, but the final reaction will be the same (that is, assuming commutativity of $|$). In this way potential reactions model *open synchronisations* which are still waiting for parties to join, whilst final reactions model *closed synchronisations* in which all parties have already joined.

This brief example showed us how we want the calculus to work: binary interactions representing potential reactions will allow more and more name parts to brought together, until we have the complete set, and a reaction can occur (when a potential reaction is committed). Now to fill in the details, and give a formal definition of the language. We will present the language somewhat indirectly in an abstraction/concretion style[1]

[1] Actually, in our context there is no distinction between abstractions and concretions as we are using the symmetric internal coordination

(as described in [87]) which separates out the processes P which define the agents of a systems, with abstractions, which $(l_1, \ldots, l_n)P_{l_1,\ldots,l_n}$ which abstract a process $P_{l_1,\ldots,l_n}$ over as of yet unspecified names $l_1, \ldots, l_n$.

3.2.1 Syntax definition

We will now define the syntax of our calculus, by defining *prefixes*, *process*, and *abstractions*. Then a process such as $x^{[2/3]}.(y)(y^{[1/3]}.\mathbf{0} + y^{[2/3]}.\mathbf{0})$ will be constructed from a prefix $x^{[2/3]}$ and an abstraction $(y)(y^{[1/3]}.\mathbf{0} + y^{[2/3]}.\mathbf{0})$ over another process $y^{[1/3]}.\mathbf{0} + y^{[2/3]}.\mathbf{0}$. What about processes such as $x^{[i/n]}(y_1, \ldots, y_m).S_{y_1,\ldots,y_m}$ where the communication prefix indicates a list of names to be received? In the abstraction/concretion approach we treat this notation as syntactic sugar for the process $x^{[i/n]}.(l_1, \ldots, l_m)S_{x_1,\ldots,}$ where the names to be received are specified as the abstracted variables in the concretion $(l_1, \ldots, l_m)S_{l_1,\ldots,l_m}$

Prefixes are then defined by the following simple grammar:

$$\pi ::= x^{[i/n]} \qquad \text{where } i, n \in \mathbb{Z} \text{ and } 1 \leq i \leq n$$

that is, the only type of prefix is a name part $x^{[i/n]}$.

Next *processes* are defined according to the grammar:

$$P, Q ::= \mathbf{0} \mid \pi_1.F_i + \ldots + \pi_n.F_n \mid P \mid Q \mid (\nu\, l_1, \ldots, l_n)P \mid D(x_1, \ldots, x_n)$$

That is, a process can be any of:

- The *null process* $\mathbf{0}$. This does exactly nothing, and represents an agent which attempts no communication actions.
- A *choice* $\pi_1.F_1 + \ldots + \pi_n.F_n$ of one of n abstractions $F_1, \ldots, F_n$ guarded by the prefixes $\pi_1, \ldots, \pi_n$. This means that the species can evolve into one of the abstractions F_i, but only after the synchronisation corresponding to the prefix π_i has occurred. We define the empty choice as $\mathbf{0}$.
- A *parallel composition* $P \mid Q$ of two processes P and Q. In this species any of the evolutions of P or Q can occur in parallel, and they can communicate with each other.
- A *name binding* or *restriction* $(\nu\, l_1, \ldots, l_n)A$ of names $l_1, \ldots, l_n$ in the species A. This prevents involving the restricted sites on this network from being exposed to the rest of the system.

- A *definition application* $D(x_1, \ldots, x_n)$, which applies the definition of the process D, supplying as arguments names $x_1, \ldots, x_n$.

The case for definition applications allows us to define the processes of a system as a list of mutually recursive definitions,

$$D(x_1, \ldots, x_n) \triangleq P$$

where $x_1, \ldots, x_n$ bind names in P. Whilst we could alternatively define recursive processes using a replication operator ! as in some treatments of the π-calculus [87], we will find recursive definitions more readable, especially when modelling chemical reactions.

Finally we define abstractions according to the following grammar:

$$F, G ::= (l_1, \ldots, l_n)A.$$

There is only one standard form of abstraction, as we list all of the abstracted names at the top level, however, we will now define operators to lift parallel composition and name restriction from the process level to the abstraction level.

Definition 3.2.2. The *parallel composition* or *colocation* of abstractions $(m_1, \ldots, m_p)P$ and $(m_1, \ldots, m_q)Q$ is defined by,

$$(l_1, \ldots, l_p)P \mid (l_1, \ldots, l_q)Q = (l_1, \ldots, l_s)(P \mid Q),$$

where $s = \max\{p, q\}$. We extend this definition to all abstractions by identifying abstractions upto α-renaming.

Definition 3.2.3. The *restriction* of names $l_1, \ldots, l_p$ in an abstraction $(m_1, \ldots, m_q)P$ is defined by,

$$(\nu\, l_1, \ldots, l_p)(m_1, \ldots, m_q)P = (m_1, \ldots, m_q)(\nu\, l_1, \ldots, l_p)P$$

where names $l_1, \ldots, l_p$ and $m_1, \ldots, m_q$ are assumed to be distinct by α-renaming.

We will also freely allow a process P to be embedded as a trivial abstraction $()(P)$, and we can see that in this case these definitions reduce to parallel composition and restriction of processes.

3.2.2 Transition semantics

We will now define an operations semantics for our language, in the form of a small-step *labelled transition system* $(\textsc{Proc}, \textsc{Abst}, \Lambda, \rightarrow)$ where $\textsc{Proc}$ is the set of processes, $\textsc{Abst}$ is the set of abstractions, and the set of labels $\Lambda = \textsc{Bag}(\mathcal{N}_{\text{part}}) \cup \{\tau\}$ will consist of either bags (multisets)[2] of part names representing potential interactions, or *silent* τ transition representing a final reaction. We will define (labelled) transition relation $\rightarrow \subseteq \textsc{Proc} \times \Lambda \times \textsc{Abst}$, using a Plotkin-style Structural Operational Semantics [96], which defines a number of transition rules based on the syntactic structure of terms.

As discussed at the start of the section, our transition rules will work by building up the parts of a name and committing the reaction when we have all of them. This motivates us to define *compatible* bags of name parts, so we can restrict ourself to potential reactions involving name parts corresponding to a single name, and *complementary* bags, so we can know when we are ready to commit a potential reactions.

Definition 3.2.4. A bag of name parts $S = \wr x^{[1/n]}, \ldots, x^{[n/n]} \wr$ corresponding to all of the name parts of a single name x is said to be *complementary*. Any subbag of a complementary bag of name parts is said to be *compatible*.

Example 3.2.5. The bag of name parts $\wr x^{[1/3]}, x^{[2/3]}, x^{[3/3]} \wr$ is complementary, whilst the bag $\wr x^{[1/3]}, x^{[3/3]} \wr$ is compatible, and the bags $\wr x^{[1/3]}, x^{[2/4]} \wr$, $\wr x^{[1/2]}, y^{[1/2]} \wr$, and $\wr x^{[1/2]}, x^{[2/2]}, x^{[2/2]} \wr$ are neither.

We now define the *commit* function, which turns an abstraction representing the products of a potential reaction, to a concrete process, representing the products of a final reaction.

Definition 3.2.6. The *committed process* resulting from an abstraction $(m_1, \ldots, m_q)A$ is defined by,

$$\text{commit}((\ell_1, \ldots, \ell_n)P) = (\nu\, \ell_1, \ldots, \ell_n)P.$$

The full set of transition rules for the calculus are then given in Figure 3.1. The rule Com combines transitions, taking the bag union of their sites, combining potential reactions if their bags of names are compatible. The combined effect of the Res, Par-Left, and Par-Right rules is to allow potential communications to build and bubble

[2]For the moment these bags will only be sets (or rather, unibags [50]), but using bags will simplify some definitions by allowing us to consider the multiset union $\uplus$, and, in all other calculi we will subsequently consider, true bags will be used as transition labels.

$$\frac{}{\sum_{i=0}^{n} \pi_i.F_i \xrightarrow{\wr\pi_j\wr} F_j}\ \textsc{Choice}_{j,n} \qquad \frac{P \equiv_\alpha P' \quad P' \xrightarrow{\alpha} P''}{P \xrightarrow{\alpha} P''}\ \textsc{Alpha}$$

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}\ \textsc{Par-Left} \qquad \frac{P \xrightarrow{\alpha} P'}{Q \mid P \xrightarrow{\alpha} Q \mid P'}\ \textsc{Par-Right}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad D(x) \triangleq P}{D(y) \xrightarrow{\alpha} P'\{y/x\}}\ \textsc{Def} \qquad \frac{P \xrightarrow{\alpha} P' \quad l_1, \ldots, l_n \notin \alpha}{(\nu\, l_1, \ldots, l_n)P \xrightarrow{\alpha} (\nu\, l_1, \ldots, l_n)P'}\ \textsc{Res}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad Q \xrightarrow{\beta} Q' \qquad \alpha \uplus \beta \text{ compatible}}{P \mid Q \xrightarrow{\alpha \uplus \beta} P' \mid Q'}\ \textsc{Com}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad \alpha \text{ complementary}}{P \mid P' \xrightarrow{\tau} \mathrm{commit}(P' \mid Q')}\ \textsc{Commit}$$

Figure 3.1: *The transition rules for the mπI-calculus.*

up to the top. However, at some point they might meet a restriction clause $(\nu\, l)$, which limits the scope of a name. The Commit rule allows a potential interaction with a complementary bag of name parts to be committed at any point.

3.2.3 Examples

We will now give some examples illustrating the use of mπI in modelling multiway interactions.

Example 3.2.7 (Encoding πI)**.** The mπI-calculus can easily encode the standard πI-calculus. To do this we map each name x and coname $\bar{x}$ to part names so

$$[\![x]\!] = x^{[1/2]} \qquad\qquad [\![\bar{x}]\!] = x^{[2/2]}.$$

If we want, we can also give a quite direct encoding of τ prefixes,

$$[\![\tau]\!] = x^{[1/1]}.$$

This then extends directly to an encoding of the whole of πI.

Example 3.2.8 (The cake pact[3])**.** Suppose John, Mary, and Peter agree to guard a delicious cake. They can only dissolve their pact by mutual agreement, however, if one of

[3]This draws on the long tradition of cake in linear logic, which includes, say [118].

them illicitly breaks the pact and eats the cake, the other two enter the outraged state. One possible definition for John's behaviour is as follows,

$$\begin{aligned}
\mathrm{John} &\triangleq \mathrm{agree}^{[1/3]}(\mathrm{dissolve},\mathrm{eat}).\mathrm{John}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\\
\mathrm{John}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat}) &\triangleq \mathrm{dissolve}^{[1/3]}.\mathrm{John}\\
&\quad + \mathrm{eat}^{[1/3]}.\mathrm{John}_{\text{full of cake}}\\
&\quad + \mathrm{eat}^{[2/3]}.\mathrm{John}_{\text{outraged}}\\
&\quad + \mathrm{eat}^{[3/3]}.\mathrm{John}_{\text{outraged}},
\end{aligned}$$

and we are also able to define agents for Mary and Peter similarly. Then John, Mary, and Peter may enter into a cake pact, via the reaction,

$$\begin{aligned}
\mathrm{John} \mid \mathrm{Mary} \mid \mathrm{Peter} \xrightarrow{\tau} \mathrm{Pact} \triangleq (\nu\,\mathrm{dissolve},\mathrm{eat})\Big(&\mathrm{John}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\\
&\mid \mathrm{Mary}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\\
&\mid \mathrm{Peter}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\Big).
\end{aligned} \tag{a}$$

How does this reaction follow from the definitions of the individual agents? Individually they can undergo potential reactions to form one third of a cake pact,

$$\mathrm{John} \xrightarrow{\wr\mathrm{agree}^{[1/3]}\int} (\mathrm{dissolve},\mathrm{eat})\mathrm{John}_{\text{with cake}}$$

$$\mathrm{Mary} \xrightarrow{\wr\mathrm{agree}^{[2/3]}\int} (\mathrm{dissolve},\mathrm{eat})\mathrm{Mary}_{\text{with cake}}$$

$$\mathrm{Peter} \xrightarrow{\wr\mathrm{agree}^{[3/3]}\int} (\mathrm{dissolve},\mathrm{eat})\mathrm{Peter}_{\text{with cake}}$$

and with two of them we could even form larger potential incomplete cake pacts, such as,

$$\begin{aligned}
\mathrm{John} \mid \mathrm{Peter} \xrightarrow{\wr\mathrm{agree}^{[1/3]},\mathrm{agree}^{[3/3]}\int} (\mathrm{dissolve},\mathrm{eat})\Big(&\mathrm{John}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\\
&\mid \mathrm{Peter}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\Big),
\end{aligned}$$

however, none of these potential cakes pacts can ever come to fruition unless they can find the third partner, so they can form a complete potential reaction,

$$\begin{aligned}
\mathrm{John} \mid \mathrm{Mary} \mid \mathrm{Peter} \xrightarrow{\wr\mathrm{agree}^{[1/3]},\mathrm{agree}^{[2/3]},\mathrm{agree}^{[3/3]}\int} (\mathrm{dissolve},\mathrm{eat})\Big(&\mathrm{John}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\\
&\mid \mathrm{Mary}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\\
&\mid \mathrm{Peter}_{\text{with cake}}(\mathrm{dissolve},\mathrm{eat})\Big),
\end{aligned}$$

with the complementary name parts $\mathrm{agree}^{[1/3]}, \mathrm{agree}^{[2/3]}, \mathrm{agree}^{[3/3]}$, which can be committed to form the final reaction (a).

Once the pact has been formed, they can either agree to dissolve it by mutual consent using $\text{dissolve}^{[1/3]}, \text{dissolve}^{[2/3]}, \text{dissolve}^{[3/3]}$,

$$\text{Pact} \xrightarrow{\tau} \text{John} \mid \text{Mary} \mid \text{Peter},$$

or one of the three can break the pact by using their $\text{eat}^{[i/3]}$ and eating all of the cake, leaving the other two outraged when they find our using the other two eat name parts,

$$\begin{aligned} \text{Pact} &\xrightarrow{\tau} \text{John}_{\text{full of cake}} \mid \text{Mary}_{\text{outraged}} \mid \text{Peter}_{\text{outraged}} \\ &\xrightarrow{\tau} \text{John}_{\text{outraged}} \mid \text{Mary}_{\text{full of cake}} \mid \text{Peter}_{\text{outraged}} \\ &\xrightarrow{\tau} \text{John}_{\text{outraged}} \mid \text{Mary}_{\text{outraged}} \mid \text{Peter}_{\text{full of cake}}. \end{aligned}$$

Example 3.2.9 (Calling Cthulhu)**.** Suppose we have an amulet of power which, when chucked into the mystic portal, will summon the dread lord Cthulhu. However, this amulet can break into 3 parts, which must be united before the summoning ritual may be performed. We can model this using processes,

$$\begin{aligned} \text{Part}_i(\text{break}) &\triangleq \text{break}^{[i/3]}.\text{BrokenPart}_i \\ &\quad + \text{summon}^{[i/4]}.\mathbf{0} \\ \text{BrokenPart}_i &\triangleq \text{unite}^{[i/3]}(\text{break}).\text{Part}_i(\text{break}) \\ \text{Portal} &\triangleq \text{summon}^{[4/4]}.\text{Cthulhu}. \end{aligned}$$

Where we represent the amulet as the complex[4],

$$\text{Amulet} \triangleq (\nu\, \text{break})(\text{Part}_1(\text{break}) \mid \text{Part}_2(\text{break}) \mid \text{Part}_3(\text{break}))$$

so it is possible for an internal reaction to break the amulet,

$$\text{Amulet} \xrightarrow{\tau} \text{BrokenPart}_1 \mid \text{BrokenPart}_2 \mid \text{BrokenPart}_3$$

and for the parts of the broken amulet to be reassembled,

$$\text{BrokenPart}_1 \mid \text{BrokenPart}_2 \mid \text{BrokenPart}_3 \xrightarrow{\tau} \text{Amulet}\ .$$

Furthermore, once we have a fully assembled amulet along with a portal, the dread lord Cthulhu may be summoned,

$$\text{Amulet} \mid \text{Portal} \xrightarrow{\tau} \text{Cthulhu}\ .$$

[4]Slightly counterintuitively, the only thing uniting the parts of the amulet is their capability to communicate privately an become broken.

Example 3.2.10 (3-reactant reactions)**.** Ultimately, all chemical reactions involve two reactants. However, in chemical modelling, reactions involving more (or even fewer) reactions are often used to abstract over the exact mechanism of the reaction, and treat it as a single atomic step. Indeed, for many chemical reactions the mechanism was only known decades after the reaction was first observed, and we cannot hope to formulate statements like "This chain of binary reactions provides the mechanism for this observed 3-reactant reaction", unless we can first describe 3-reactant reactions as atomic, first class events in their own right. Hence, we are motivated to have a formalism for chemical reaction capable of describing reaction with any number of reactants and products.

Our mπI-calculus is exactly what is required to express multi-reactant reactions. For example, consider the following reaction,

$$R_1 + R_2 + R_3 \longrightarrow P$$

where three reactants R_1, R_2, R_3 combine to make the product P. We are able to express this as,

$$R_1 \triangleq r^{[1/3]}.P \qquad R_2 \triangleq r^{[2/3]}.\mathbf{0} \qquad R_3 \triangleq r^{[3/3]}.\mathbf{0},$$

or, by relying on dynamic complex formation to form P from components,

$$R_1 \triangleq r^{[1/3]}(l).R_1^*(l) \qquad R_2 \triangleq r^{[2/3]}(l).R_2^*(l) \qquad R_3 \triangleq r^{[3/3]}(l).R_3^*(l),$$

where we define P as the dynamic complex,

$$P \triangleq (\nu\, l)(R_1^*(l) \mid R_2^*(l) \mid R_3^*(l)).$$

3.2.4 Symmetry of communication

One of the nicest properties of the πI-calculus was that removing the syntactic between receiving and sending (via names and conames respectively) lead them these concepts to be formally dual, meaning that they replacing all sends with receives does not change the meaning of a process [105]. The concept of *duality* does not quite apply to our multiway calculus as whilst before a channel has two ends – the name and coname – it can now have many represented by its name parts. Nevertheless, we are able to generalise this property with the following *symmetry* property.

Theorem 3.2.11 (Symmetry theorem)**.** *Let T be a map that acts on a process P by applying a permutation σ_n to the indices $[1/n], [2/n], \ldots, [n/n]$ of the part names in P for each n. Then we have that,*

$$P \xrightarrow{\alpha} P' \qquad \Leftrightarrow \qquad T_\sigma P \xrightarrow{T_\sigma \alpha} T_\sigma P' \ .$$

Proof. We can check that each of the transition rules of mπI is unaffected by such a map T. The result then follows by induction on the structure of derivations. □

3.3 Compatibility networks and the bond-calculus

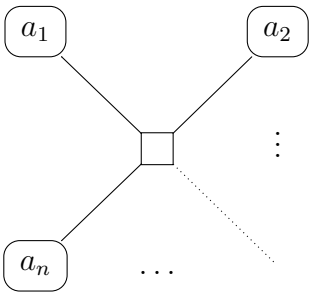
Continuous π does not have a notion of names and coname, but rather, defines the reactivity between names using an *affinity network*, a graph with edges between compatible names, along with labels containing quantitative information about the rate of reactions. This provides a more intuitive framework for modelling biochemical reactions agents have reaction sites which can be involved in a range of different reactions at different rates. We will now attempt to span the gap between this approach (ignoring the qualitative aspect for the moment) and the more minimal mπI-calculus, to develop the bond-calculus, a richer qualitative calculus for multiway interactions.

3.3.1 Introducing compatibility networks

For this we will define a *compatibility network* which takes the qualitative part of affinity but generalises it to multiway interactions.

Definition 3.3.1 (Compatibility network)**.** A *compatibility network* $\mathcal{C}$ is a multi-hypergraph on names. That is, $\mathcal{C}$ consists of a set $V_{\mathcal{C}} = \mathcal{S}$ of nodes and a set $E_{\mathcal{C}} \subset \textsc{Bag}(\mathcal{S})$ of bags of sites representing hyperedges.

A hyperedge in the compatibility network,



indicates that the names $a_1, \ldots, a_n$ are complementary and will react together. We are able to use to give a generalised definition of complementarity and compatibility.

Definition 3.3.2. A bag of names $S = \wr m_1 \cdot a_1, \ldots, m_n \cdot a_n \int$ is said to be *complementary* with respect to the compatibility network $\mathcal{C}$ if there exists a multihyperedge in $\mathcal{C}$ with exactly these nodes and multiplicities. Any subbag of a complementary bag (with respect to $\mathcal{C}$) of name parts is said to be *compatible* (with respect to $\mathcal{C}$.

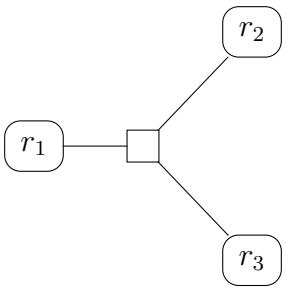
We can see that in the case that if each node of $\mathcal{C}$ is involved in exactly one hyperedge (and is involved exactly once), we could capture this notion of compatibility in mπI by setting $a_1 = a^{[1/n]}, \ldots, a_n = a^{[n/n]}$. However, in general it will useful to make use of the same name in multiple reactions, and also to use have a single name being involved in a hyperedge more than once to represent how many copies of the site are required for the reaction to go ahead.

Now, if we forget about the distinction between names and name parts, and use these new definitions in the transition rules in Figure 3.1, we get a new generalised calculus, with the compatibility network $\mathcal{C}$ determining what multiway interactions can be formed.

Example 3.3.3 (3-reactant reactions)**.** We are now able to encode the same 3 reaction we considered before more idiomatically, by defining agents,

$$R_1 \triangleq r_1.P \qquad R_2 \triangleq r_2.\mathbf{0} \qquad R_3 \triangleq r_3.\mathbf{0},$$

and compatibility network,



Example 3.3.4 (Repeated reactant)**.** Suppose multiple copies of a reactant are involved in a reaction, e.g.

$$A + 2B \longrightarrow P$$

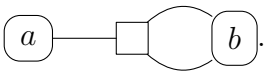
In mπI (and all the more so in other variants of the π-calculus) we would be forced to introduce some degree of asymmetry or duplication into our definitions, giving B two sites, and forcing each copy of B to make an arbitrary choice as to which role it plays, so for example, using the definitions,

$$A \triangleq r^{[1/3]}.P \qquad B \triangleq r^{[2/3]}.\mathbf{0} + r^{[3/3]}.\mathbf{0}.$$

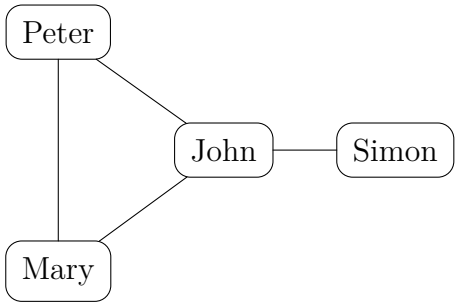
However, we are now able to give a much more natural definition of B, using a single site, in

$$A \triangleq a.P \qquad B \triangleq b.\mathbf{0}.$$

using the compatibility network,



Example 3.3.5 (Collaboration networks)**.** Research collaborations occur when researchers with complementary skill come together together to work on a single project. In the social sciences, there has been considerable interest in analysing networks of collaborating researchers, with nodes representing researchers, and edges representing collaborations [89, 67, 9]. For example, we can represent a network of collaborations between John, Mary, Peter, and, Simon via the following network,

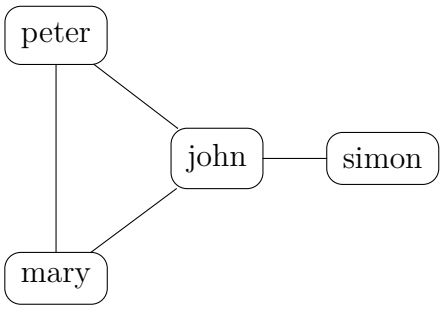


.

We are able to represent the formation of research collaborations using the following processes:

$$\text{John} \triangleq \text{john.John}^*(l) \qquad \text{Mary} \triangleq \text{mary.Mary}^*(l)$$
$$\text{Peter} \triangleq \text{peter.Peter}^*(l) \qquad \text{Simon} \triangleq \text{simon.Simon}^*(l)$$

and an affinity network reflecting the structure of the collaboration network,

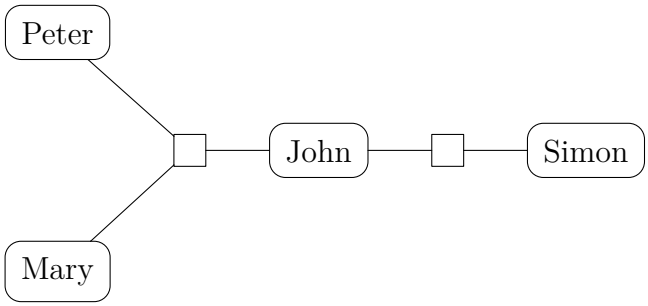


.

In this system complexes can form corresponding to each collaboration in the collaboration network. This models collaborations as binary synchronisations, each involving a pair of individuals.

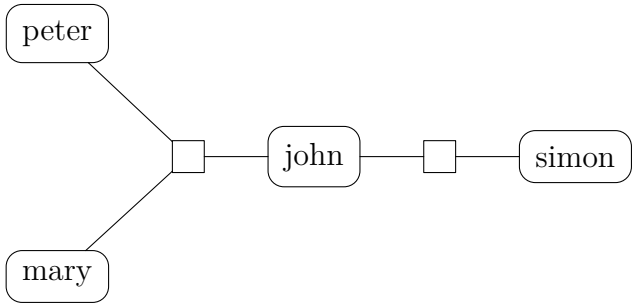
However, graphs do not fully capture the nature of research collaboration, since many collaborations involve groups of more than two people, and representing this as binary collaborations does not capture the multiparty groups involved. Instead, hypergraphs

have been proposed as better models of collaboration [107, 111], with node representing researchers, and hyperedges representing collaborations. So, for example, the previous collaboration graph could be explained by the following collaboration hypergraphs,



.

It would be difficult to represent this network in process algebras without multiway synchronisation, as it would be necessary to specify the order in which collaborations are formed. However, using multiway synchronisation, we are able to specify this scenario via the compatibility network,



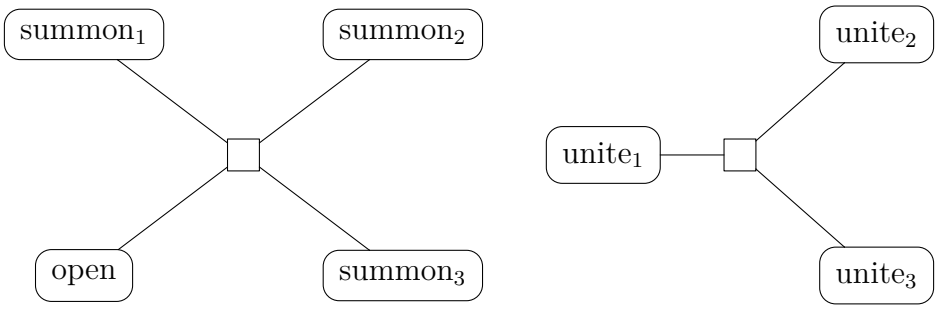
.

3.3.2 The limitations of static networks

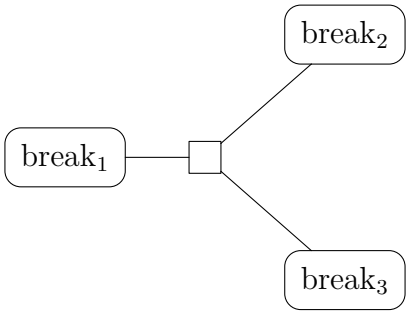
As we have just seen, compatibility networks give a concise way to describe the potential reactions between agents based on reusable sites. Compared with names and conames or the more general part names, compatibility networks give a nice visual representation of the sites underlying the reactions, however, as currently defined, the limit the range of potential reactions given they are static, and do not include new names which are introduced as the process evolves. As an example, consider our Cthulhu example. We might like to describe the agents as,

$$\begin{aligned}
\mathrm{Part}_i(\mathrm{break}) &\triangleq \mathrm{break}.\mathrm{BrokenPart}_i \\
&\quad + \mathrm{summon_i}.\mathbf{0} \\
\mathrm{BrokenPart}_i &\triangleq \mathrm{unite}_i(\mathrm{break}_1, \mathrm{break}_2, \mathrm{break}_3).\mathrm{Part}_i(\mathrm{break}_i) \\
\mathrm{Portal} &\triangleq \mathrm{open}.\mathrm{Cthulhu},
\end{aligned}$$

however, what about the compatibility network? Well, we can describe the summoning ritual, and reuniting the parts of the amulet using the network,



and would like to describe the breaking of the amulet using the network,



However, this does not make sense in the context of the global compatibility network, as the sites involved ($\text{break}_1, \text{break}_2, \text{break}_3$) are private; as we allow α-conversion of bound names, these site names are just place holders, and so if we include them in the affinity network, the structural congruence of the language can freely rename these sites, which would alter the behaviour of the agent. So, as it stands, we have no way of allowing internal reactions to happen within dynamically formed complexes.

This problem was resolved in continuous π using *local affinity networks*, which allow name binders to specify a local affinity network applying to the names within their scope. For example, we would represent an amulet containing only two parts using agents,

$$\text{Part}_1 \triangleq \left(\nu \boxed{\text{break}_1} \text{---} \boxed{\text{break}_2}\right) \text{unite}_1\langle \text{break}_2 \rangle.\text{BrokenPart}_1(\text{break}_1)$$

$$\text{Part}_2 \triangleq \text{unite}_2(\text{break}_2).\text{BrokenPart}_2(\text{break}_2)$$

In this model one of the parts defines a local affinity network which is to govern the interactions in the complex, and sends the a site name to the other component, allowing interaction between them after communication has occurred. Whilst this proved quite effective, it has the undesirable effect of coupling the definition of the affinity networks, to the structure of the agents, leading to more complex models with embedded local affinity networks for each local scope. Moreover, this solution relies fundamentally on

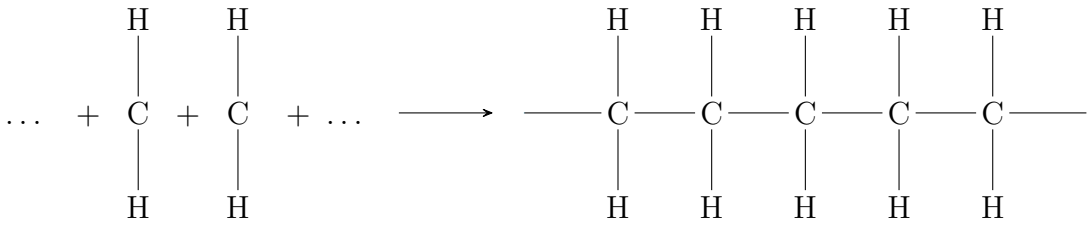


Figure 3.2: *Polyethene chain formation*

the asymmetry between send and receive in the π-calculus, leading to asymmetric for the different parts of the amulet, whilst, in principle they should have the same structure. That is, one of the parts is expected to do the sending and define the local affinity network, whilst another does the receiving. All of this breaks down when we add in n-way internal synchronisation; if n processes bind bind together, it does not make sense for a single one of them to define the affinity network governing the internal reactions for the whole complex.

In the next subsection we will present a better solution, splitting the concept of names into *sites* and *locations* to define the bond-calculus. This will allow for processes to be described using a fixed, finite set of sites, whose reactivity is specified by a static compatibility network, which agent may freely create new mobile locations, allowing sites to be localised to within a complex. This will both alleviate the limitations of a static compatibility network, and present opportunities for modelling a wide range of new agents.

3.3.3 From names to sites and locations

As we have seen, we have a problem using a compatibility network to specify the reactivity of sites, as the static nature of the network cannot adapt to include names which are generated dynamically as the process evolves. In order to understand what the static affinity structure represents, we can consider the formation of polymers. Polymers form when many molecules (called monomers) bind together into chains of arbitrary length. For example, Polyethene is formed as chains of Ethene (CH_2) monomers. We could represent this in the πI-calculus using agents,

$$\begin{aligned} \mathrm{CH}_2 &\triangleq (\nu\, s)\,(\mathrm{Site}_s \mid \mathrm{Site}_s) \\ \mathrm{Site}_s &\triangleq \mathrm{join}(\mathrm{unjoin}).\mathrm{Site}^*_{s,\mathrm{unjoin}} + \overline{\mathrm{join}}(\mathrm{unjoin}).\mathrm{Site}^*_{s,\overline{\mathrm{unjoin}}} \\ \mathrm{Site}^*_{\mathrm{s,unjoin}} &\triangleq \mathrm{unjoin}.\mathrm{Site}_s, \end{aligned}$$

where we form CH_2 molecules as a parallel composition of two reaction sites, which are able to bond to the reaction sites on other molecules (or even to the original molecule, forming a double bond)[5]. For example, we can form a chain of length 3 as follows,

$$\begin{aligned}
CH_2 \mid CH_2 \mid CH_2 &\to (\nu\, s_1)\,(\mathrm{Site}_{s_1} \mid \mathrm{Site}_{s_1}) \mid (\nu\, s_2)\,(\mathrm{Site}_{s_2} \mid \mathrm{Site}_{s_2}) \mid CH_2 \\
&\to (\nu\, s_1, s_2)\left(\mathrm{Site}_{s_1} \mid (\nu\, \mathrm{unjoin}_1)\left(\mathrm{Site}^*_{s_1,\mathrm{unjoin}_1} \mid \mathrm{Site}^*_{s_2,\overline{\mathrm{unjoin}_1}}\right) \mid \mathrm{Site}_{s_2}\right) \mid CH_2 \\
&\to (\nu\, s_1, s_2, s_3)\Big(\mathrm{Site}_{s_1} \mid (\nu\, \mathrm{unjoin}_1)\left(\mathrm{Site}^*_{s_1,\mathrm{unjoin}_1} \mid \mathrm{Site}^*_{s_2,\overline{\mathrm{unjoin}_1}}\right) \\
&\qquad \mid (\nu\, \mathrm{unjoin}_2)\left(\mathrm{Site}^*_{s_2,\mathrm{unjoin}_2} \mid \mathrm{Site}^*_{s_3,\overline{\mathrm{unjoin}_2}}\right) \mid \mathrm{Site}_{s_3}\Big).
\end{aligned}$$

We can see that the private sites $\mathrm{unjoin}_1, \mathrm{unjoin}_2$ are used to establish the links of the polymer, and that reactions between these sites allow the chain to break up. However, chains can only break at between two adjacent monomers, so each unjoin site can only react with corresponding $\overline{\mathrm{unjoin}}$ site. Since polymers of arbitrary length may form, we have infinitely many potential unjoining points, and hence our model fundamentally requires an infinite supply of distinct channels, making describing their reactivity via a static compatibility network seem rather infeasible.

We should however ask ourselves whether this system actually involves infinitely many distinct sites? Well, no – whilst we may have arbitrarily many different bonds, the basic units involved in each (the polyethene monomers) are the same. That is, all of the different unjoin sites are really the same chemical site – what distinguishes their ability to react is not their structural compatibility, but their location within the polymer. Therefore, in order to successfully this kind of system, we are motivated to define a new calculus, the bond-calculus, which splits the concept of channel names in two, the *site* which governs which sites are structurally compatible based on the affinity network[6], and the *location* which specifies where the site is and restricts reactions to occurring between sites at the same location. The locations can by created dynamically and passed between processes (so they assume the role of the mobile channels in πI), whilst the site names are static, and predefined by the compatibility network.

3.3.4 Syntax and semantics

In the bond-calculus, we have a set $\mathcal{S}$ of *site names*, and a set $\mathcal{L}$ of *location names*. A compatibility network will now apply to the site names of the calculus.

[5]This is a quite rough approximation of how polythene polymerisation actually works, but could be extended into a more chemically accurate model.

[6]Biochemically speaking, this represents the *conformity* of the reaction site.

Definition 3.3.6. A *compatibility network* $\mathcal{C}$ is a multi-hypergraph on site names.

We will now define the syntax rules for the language. *Prefixes* are defined by the following grammar:

$$\pi ::= \ s \ \mid \ s@\ell$$

that is, sites consist of an unlocated site name s, or a located site name $s@\ell$, representing s located at ℓ.

The grammar for *processes* is as follows:

$$P, Q ::= \ \mathbf{0} \ \mid \ \pi_1.F_1 + \ldots + \pi_n.F_n \ \mid \ P \,|\, Q \ \mid \ (\nu\, \ell_1, \ldots, \ell_n)P \ \mid \ D(s_1, \ldots, s_n; \ell_1, \ldots, \ell_m)$$

This definition is much the same as in mπI, except restrictions now involve location names rather than channel names, and definition applications now take two lists of parameters: one for sites $s_1, \ldots, s_n$ and one for locations $\ell_1, \ldots, \ell_n$.

Finally, we define abstractions according to the following grammar,

$$F, G ::= \ (\ell_1, \ldots, \ell_n)A.$$

Parallel composition and location restriction on abstractions are defined in exactly the same way as in Section 3.2.

We will now define the labelled transition system specifying the operation semantics for the language. This time the labels will be neither sites or locations, but *located sites* $s@\ell$ consisting of both a site s and a location ℓ. We may, however, treat unlocated sites a special case of located sites, using the top location $\top$, so s can considered shorthand for $s@\top$.

We then have the following definition of compatibility, which now requires sites to be at the same location in order to react.

Definition 3.3.7. A bag of located sites $S = \wr m_1 \cdot a_1@\ell_1, \ldots, m_n \cdot a_n@\ell_n \wr$ is said to be *complementary* with respect to the compatibility network $\mathcal{C}$ if there exists a multi-hyperedge in $\mathcal{C}$ with exactly these sites and multiplicities, and if $\ell_1 = \ell_2 = \ldots = \ell_n$. Any subbag of a complementary bag (with respect to $\mathcal{C}$) of located sites is said to be *compatible* (with respect to $\mathcal{C}$.

We must also define the locations function, which gives all the locations in a bag of located sites.

$$\frac{}{\sum_{i=0}^{n} \pi_i.F_i \xrightarrow{\wr \pi_j \wr} F_j}\ \textsc{Choice}_{j,n} \qquad \frac{P \equiv_\alpha P' \quad P' \xrightarrow{\alpha} P''}{P \xrightarrow{\alpha} P''}\ \textsc{Alpha}$$

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}\ \textsc{Par-Left} \qquad \frac{P \xrightarrow{\alpha} P'}{Q \mid P \xrightarrow{\alpha} Q \mid P'}\ \textsc{Par-Right}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad D(\mathbf{x};\mathbf{l}) \triangleq P}{D(\mathbf{y};\mathbf{m}) \xrightarrow{\alpha} P'\{\mathbf{y}/\mathbf{x},\mathbf{m}/\mathbf{l}\}}\ \textsc{Def} \qquad \frac{P \xrightarrow{\alpha} P' \quad l_1,\ldots,l_n \notin \text{locations}(\alpha)}{(\nu\, l_1,\ldots,l_n)P \xrightarrow{\alpha} (\nu\, l_1,\ldots,l_n)P'}\ \textsc{Res}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad Q \xrightarrow{\beta} Q' \qquad \alpha \uplus \beta \text{ compatible}}{P \mid Q \xrightarrow{\alpha\uplus\beta} P' \mid Q'}\ \textsc{Com}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad \alpha \text{ complementary}}{P \mid P' \xrightarrow{\tau} \text{commit}(P' \mid Q')}\ \textsc{Commit}$$

Figure 3.3: *The transition rules for the mπI[$\mathcal{C}$]-calculus.*

Definition 3.3.8. We define the *locations contained in a bag of sites* inductively as,

$$\begin{aligned}\text{locations}(\alpha \uplus \beta) &= \text{locations}(\alpha) \cup \text{locations}(\beta)\\ \text{locations}(\wr a@\ell \wr) &= \{\ell\}\\ \text{locations}(\wr a \wr) &= \varnothing\end{aligned}$$

where α and β are bags of sites, a is a site name, and ℓ is a location name.

The transition rules of the mπ[$\mathcal{C}$]-calculus are now given in Figure 3.3. These are almost the same as those we gave in Figure 3.1 for mπI, except they use sites and locations, and our new definitions of complementarity/compatibility.

3.3.5 Examples

We will now give a number of examples demonstrating the bond calculus, and see that we are now able to use compatibility networks to intuitively model a range of systems involving dynamic locations.

Example 3.3.9 (Polymers)**.** In the bond-calculus, the compatibility network now gives a static network of compatibility between sites, but this can be extended into a dynamic network of interactions between agents when they are combined with locations. This now

allows us to model our polymer system with agents,

$$\begin{aligned}\mathrm{CH}_2 &\triangleq (\nu\, m)\,(\mathrm{Site}_m \mid \mathrm{Site}_m)\\ \mathrm{Site}_m &\triangleq \mathrm{join}(\ell).\mathrm{Site}^*_{m,\ell}\\ \mathrm{Site}^*_{\mathrm{m},\ell} &\triangleq \mathrm{unjoin}@\ell.\mathrm{Site}_m,\end{aligned}$$

where ℓ, m both represent locations and join, unjoin are the sites (the Site agents both encapsulate one of the reaction sites and its two possible states, but are just agents, not the actual sites), combined with the compatibility network,

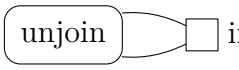
In this model, we use the compatibility network to capture that, rather than a complementary join/$\overline{\text{join}}$ coname pair, binding is between two identical join sites, simplifying the definition of the agents, and removing a needless asymmetry from the model.

This allows monomers to dynamically bond together much as before, so for example, the complex CH_2=CH_2 could be represented by a process,

$$(\nu\, m_1, m_2)\left(\mathrm{Site}_{m_1} \mid (\nu\, \ell)\left(\mathrm{Site}^*_{m_1,\ell} \mid \mathrm{Site}^*_{m_2,\ell}\right) \mid \mathrm{Site}_{m_2}\right)$$

Now if we look in detail, inside this chain, the link occurs in the process,

$$(\nu\, \ell)\left(\mathrm{Site}^*_{m_1,\ell} \mid \mathrm{Site}^*_{m_2,\ell}\right) = (\nu\, \ell)\,(\mathrm{unjoin}@\ell.\mathrm{Site}_{m_1} \mid \mathrm{unjoin}@\ell.\mathrm{Site}_{m_2})$$

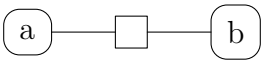
However, since we have a multi-hyperedge 

in the compatibility network, and each of the unjoin sites in this process are located at the same location ℓ, this process can transition to,

$$(\nu\, \ell)\,(\mathrm{Site}_{m_1} \mid \mathrm{Site}_{m_2}) \equiv \mathrm{Site}_{s_1} \mid \mathrm{Site}_{m_2}$$

meaning that the whole CH_2=CH_2 polymer is allowed to break back down it $\mathrm{CH}_2 \mid \mathrm{CH}_2$. This shows that, with a simple compatibility network, it is possible to specify an infinite number of polymer interactions, whilst reflecting the symmetry of the underlying chemical interactions.

Example 3.3.10. The reader may well ask whether the fixed static, compatibility network still restricts the calculus in some way, compared to more conventional names and conames? In fact, this is not the case as we are able encode πI, using a simple two node compatibility network,

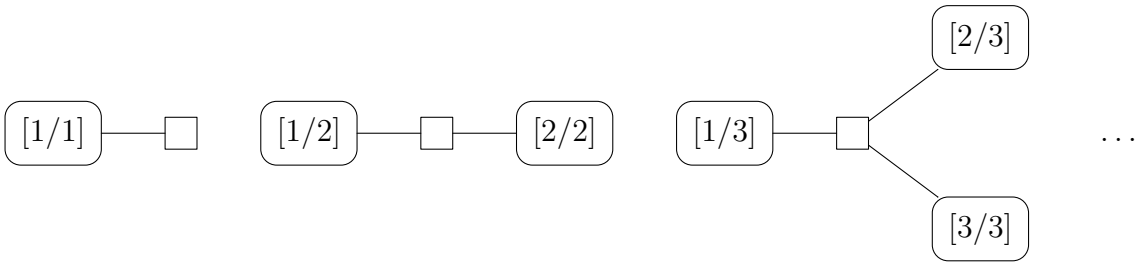


Then, setting the location set as $\mathcal{L} = \mathcal{N}$, we are able to encode names and coname prefixes as follows,

$$[\![x]\!] = a@x \qquad\qquad [\![\overline{x}]\!] = b@x$$

This then extends directly to an encoding of the whole calculus.

Example 3.3.11. We are also able to encode mπI processes, however, this requires one multi-edge in the compatibility network, for each degree of multiway synchronisation in the program (so, unless we allow an infinite compatibility network, this translation will require inspecting the whole program to determine the size of compatibility network required):



Then, setting the location set as $\mathcal{L} = \mathcal{N}$, we are able to encode name part prefixes as follows,

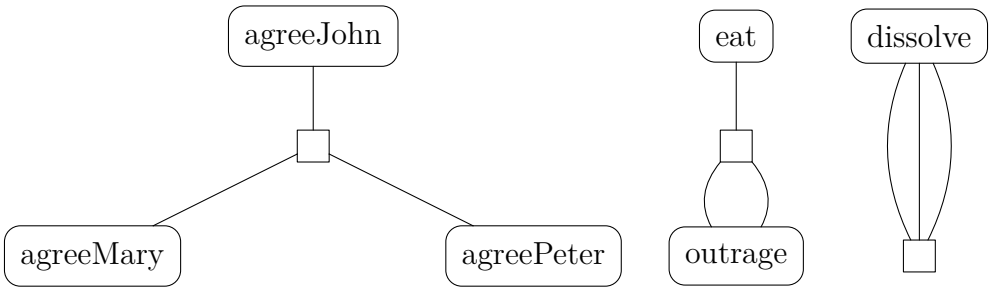
$$[\![x^{[i/n]}]\!] = [i/n]@x$$

This then extends directly to an encoding of the whole calculus.

Example 3.3.12 (The cake pact)**.** We will now try to model our cake pact example using a compatibility network. We define the agent for John as follows,

$$\begin{aligned} \text{John} &\triangleq \text{agreeJohn}(\ell).\text{John}_{\text{with cake},\ell} \\ \text{John}_{\text{with cake},\ell} &\triangleq \text{dissolve}@\ell.\text{John} \\ &\quad + \text{eat}@\ell.\text{John}_{\text{full of cake}} \\ &\quad + \text{outrage}@\ell.\text{John}_{\text{outraged}}, \end{aligned}$$

and express the other agents similarly. We then combine this with the compatibility network,

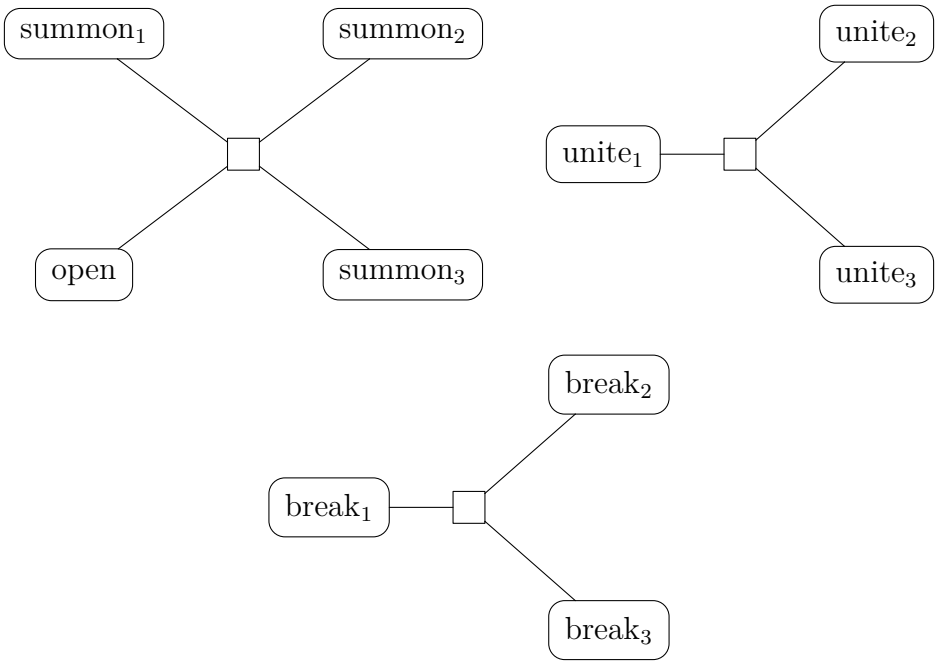


This network allows us to clearly represent that forming a pact explicitly requires the participation of each of John, Mary, and Paul, whereas the scenarios where someone eats the cake, or they dissolve the pact are symmetry, respectively requiring one eat action and two outrage, or three dissolve action, regardless of which agents within the pact supplies them.

Example 3.3.13 (Calling Cthulhu)**.** We can now describe the summoning ritual using the agents,

$$\begin{aligned} \mathrm{Part}_{i,\ell} &\triangleq \mathrm{break}_i@l.\mathrm{BrokenPart}_i \\ &\quad + \mathrm{summon}_i.\mathbf{0} \\ \mathrm{BrokenPart}_i &\triangleq \mathrm{unite}_i(\ell).\mathrm{Part}_{i,\ell} \\ \mathrm{Portal} &\triangleq \mathrm{open.Cthulhu}, \end{aligned}$$

Now we have separated sites and locations, the following compatibility network suffices to describe their interactions,



Example 3.3.14 (Reusable reactants)**.** The reader may be wondering why we allowed abstractions with different numbers of abstracted names to be collocated. We argue that if we only allowed composing abstractions with the same number of names this would unduly couple the models of each components of a reaction. For example, suppose we want to model the reaction,

$$A + B + E \longrightarrow AB + E$$

where enzyme E allows A and B to bond to form complex AB. We could define E as $E \triangleq e(\ell).E$, and model the products of this reaction being formed as the colocation,

$$(\ell)A_\ell \mid (\ell)B_\ell \mid (\ell)E = (\ell)(A_\ell \mid B_\ell \mid E)$$

however, there is no reason why the same site of E could not be involved in other reactions, for example,

$$A + B + C + D + E \quad \longrightarrow \quad AB + CD + E$$

where we would have to define E as $E \triangleq e(\ell, m).E$ so the products could be formed via the colocation,

$$(\ell, m)A_\ell \mid (\ell, m)B_\ell \mid (\ell, m)C_m \mid (\ell, m)D_m \mid (\ell, m)E = (\ell, m)(A_\ell \mid B_\ell \mid C_m \mid D_m \mid E)$$

It makes more sense to give a single definition of E as $E \triangleq e.E = e().E$ and use it in both of the colocations as,

$$\begin{aligned}(\ell)A_\ell \mid (\ell)B_\ell \mid ()E &= (\ell)(A_\ell \mid B_\ell \mid E)\\ (\ell, m)A_\ell \mid (\ell, m)B_\ell \mid (\ell, m)C_m \mid (\ell, m)D_m \mid ()E &= (\ell, m)(A_\ell \mid B_\ell \mid C_m \mid D_m \mid E).\end{aligned}$$

Committing these will give the correct products of each of the two reactions,

$$\begin{aligned}\text{commit}((\ell)(A_\ell \mid B_\ell \mid E)) &= (\nu\, \ell)(A_\ell \mid B_\ell \mid E)\\ \text{commit}((\ell, m)(A_\ell \mid B_\ell \mid C_m \mid D_m \mid E)) &= (\nu\, \ell, m)(A_\ell \mid B_\ell \mid C_m \mid D_m \mid E)\end{aligned}$$

These processes represent two and three different molecules respectively, although this may not be obvious from looking at them. We will see how to easily distinguish the molecules resulting from a reaction in Section 4.5 when we consider *normal forms*.

Comparing these examples to those shown previously, we see that the simplicity of mπI, and its ability to capture arbitrary degrees of n-way synchronisation directly in the syntax of the agents was an advantage, however, we once add locations and compatibility networks to get the mπI@-calculus, we are still able to capture the full range of behaviour, and the models become significantly more readable. As we now move to the next section, we will see how when we are given the task of assigning quantitative rates to our reactions, the advantages of using a network to express reactivity are compounded, allowing us to introduce *affinity networks* which also capture the reactions rates, and establish a clean separation between the behaviour of agents and their interaction dynamics.

Chapter 4

Quantitative Modelling and General Kinetics

This chapter will introduce a new calculus for modelling quantitative biological systems, the continuous bond-calculus. This will extend the qualitative calculus we developed in the previous chapter, to a two level calculus for quantitative mixtures of species, with full support for general kinetics. First we will give a little background on the problem of quantitative modelling and general kinetics in process algebra. Then we will how affinity networks can be extended to model general kinetics, and introduce a two level calculus for quantitative mixtures of different species of process. Next we will define the formal syntax for the calculus and an operational semantics for species, and then consider define a structural congruence so we can tell when two species are the same, and a normal form so we can give a unique form to each species. Next comes the process level vector field semantics, which represents the spaces of possible mixtures of species as a vector space, and defines the dynamics for the language. Finally, we will discuss the implementation we have developed for the language.

4.1 Background

The process calculi we have considered thus far have all used non-deterministic semantics, which models the passage of time as sequences of communication actions, abstracting away the quantitative rates which determine the exact temporal evolution of the system, along with the probabilistic nature of its behaviour. This abstraction usually makes sense when modelling computer systems (with the notable exception of real time systems), since most programs avoid making assumptions about the exact timing of events (with varying

degrees of success), ensuring their correct behaviour via explicit sequencing of events in sequential programming, and via synchronisation at appropriate points of the program; timing dependent behaviour is usually considered a bug, rather than an intended feature.

In biological systems, things are very different, since there is little built in central synchronisation[1], and so the function of many biological mechanisms depends crucially on the timing of events. For example, in many organisms, biochemical oscillators are used to establish circadian clocks, which may be used, for example, synchronise the production of chlorophyll to the cycle of the sun. The exact quantitative rates of the reactions are vital to producing an accurate and robust clock. In these systems and many others, the precise rates governing the evolution of the system are not just a potential source of errors, but rather, an integral part of the design of the system, directly exploited to implement its behaviour.

So, how can process calculi model these aspects of the system? Various stochastic process calculi have been developed which augment the transitions between processes with quantitative rates, allowing the evolution of the process to be simulated stochastically. For example, the Stochastic-π calculus, Stochastic β-binders, and Bio-PEPA are all specified in terms of labelled transition systems, with quantitative rates (which correspond to the parameters of exponential distributions), allowing the system to be translated to a Continuous Time Markov chain, and simulated via Guillespie's Stochastic Simulation Algorithm. These techniques have been successfully applied to modelling a wide variety of stochastic systems including – citations here.

A limitation of these techniques is that the difficulty of simulation increases exponentially with the number of agents involved. Given many of the systems we are interested in modelling involve interactions between millions of different molecules or cells, stochastic simulation soon becomes intractable. There is, however, another approach based on modelling the population of each distinct species of agents as a continuous (real valued) variable, and then using the rates of change of each species to determine a system of differential equations determining the evolution of the system. This approach has been pursued by Continuous-π and the fluid flow approximation for PEPA and Bio-PEPA. Using these deterministic continuous-space models makes it possible to efficiently simulate systems with arbitrarily large numbers of agents, however, they are only accurate in the limit of large populations, and break down if the population of a species is too low, or if

[1]Certain systems such as circadian cycles and axons in the heart do provide some degree of synchronisation, however, these are built out of unsynchronised, time dependent processes, via feedback cycles.

chaotic effects are prevalent.

The continuous interpretation of process algebra corresponds to conventional practice in modelling chemical reaction kinetics, where reactions are directly represented as differential equations, with variables representing species. The basic assumption in these models is that reactions are governed by the *law of mass action*, which states that the rate of reaction is directly proportional to the concentration of each reactant, scaled by a constant of proportionality, the *stoichiometric rate* for the reaction. This is in theory enough to describe all chemical process, since, we know all chemical reactions can be broken down into a sequence of atomic (binary) reactions, and it can be seen that these follow the law of mass action[2]. Since many process algebras for biochemical modelling such as continuous-π and stochastic-π assume the law of mass action as the basis of their quantitative semantics, they require all reaction to be broken down into chains of mass action reactions to be modelled.

However, when observe the rates of many actual chemical reactions, they do not follow the law of mass action, but instead appear to obey a more complicated nonlinear rate law, for example, the enzymatic reactions which are so frequent in biological systems, are usually modelled by the Michaelis-Menten kinetic law. These general kinetic laws will will only ever be imperfect approximations of the underlying reaction mechanism, however, in many situations they give good agreement with experimental data, and their use is ubiquitous throughout models of chemical kinetics.

The wide use of general kinetic laws means that many biological models are difficult to replicate in many process calculi, without translating them into mass action models, and performing additional costly experiments to estimate the additional parameters introduced by the mass action model. Bio-PEPA addressed this problem by allowing models to define additional functional rate laws which an be used to determine the rate of transitions, allowing it to successfully model a variety of systems using general kinetics. It was not, however, obvious how this approach could be applied to the compositional vector field semantics of continuous-π, as this semantics relies closely on the linearity of the mass action kinetic law to combine the actions of different processes in a compositional way.

In this section we will attempt to build a quantitative process calculus for biochemical modelling, combining both the new linguistic features of multiway internal coordination

[2]This follows as each pair of molecules in a given step of the reaction will react together at the same rate (there can be no other interactions with the other molecules since this we assume there are no intermediate reactions) which gives the stoichometric rate, and hence the rate of the overall reaction will scale with the number of pairs of molecules, leading directly to the law of mass action.

and sites/locations, with a compositional continuous state semantics, via *affinity networks* which extend compatibility networks to include functional rate laws. Our semantics extends the vector field semantics introduced by continuous-π, to include general kinetics, as described by functional rate laws. We will see that this combination of multiway synchronisation, affinity networks, and kinetic laws, provides an expressive and flexible framework for biochemical modelling.

4.2 From reaction rates to affinity networks

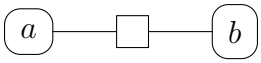
Suppose we are given a chemical reaction,

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

in which reactants A and B bond to form a complex C at stoichiometric rate r. We could model the quantitative aspects of this system with agent definitions,

$$A \triangleq a(\ell).A^*_\ell \qquad B \triangleq b(\ell).B^*_\ell \qquad C \triangleq (\nu\,\ell)\,(A^*_\ell \mid B^*_\ell)$$

and compatibility network,

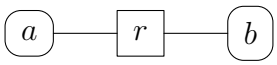


however, where do we include the quantitative rate of the reaction?

Well, whilst for non-deterministic systems we use compatibility networks to specify the capability for nodes to react, for quantitative systems we introduce *affinity networks*, which also specify the rate of the reaction, by labelling (multi)edges with the stoichiometric rate for the reaction.

Definition 4.2.1 (Stoichiometric affinity network)**.** An *(stoichiometric) affinity network* is a multihypergraph on sites, with hyperedges labelled with stoichiometric rates $r \in \mathbb{R}$.

We should then be able to turn our example into a quantitative model by specifying the affinity network,



Note that in contrast to other quantitative process algebras, the affinity network abstracts the qualitative behaviour of agents from the quantitative rates of reactions. This means the same model can be interpreted as either a qualitative or a quantitative model or

reinterpreted with different rates, by changing the network used, without needing to change the definition of agents.

This works well for mass action reactions, but what about those governed by other kinetic laws? For these we need to define rate laws, which take the concentration of each reactant, and give the reaction rate as $r \in \mathbb{R}$. Since these rate laws are often parametrized with one or more rate constants to match the specific reaction being modelled, we represent them as curried functions $\mathfrak{F} : \mathbb{R}^* \to \mathbb{R}^* \to \mathbb{R}$, which take as their first argument a list of real valued rate constants, and as their second a list of concentrations, to return a real reaction rate.

Definition 4.2.2 (Rate law)**.** A *rate law* $F : \mathbb{R}^* \to \mathbb{R}$ returns a reaction rate $r \in \mathbb{R}$, given a list of reactant concentrations $\mathbf{x} \in \mathbb{R}^*$.

Definition 4.2.3 (Rate law family)**.** A *rate law family* $\mathfrak{F} : \mathbb{R}^* \to \mathbb{R}^* \to \mathbb{R}$ returns a rate law $F_{\mathbf{k}} : \mathbb{R}^* \to \mathbb{R}$, given a list of rate constants $\mathbf{k} \in \mathbb{R}^*$.

We will now look to generalise affinity networks to allow edges to be labelled with the rate laws for each reaction. One thing we must take into account, is that these generalised affinity networks will not quite be (multi)hypergraphs, since, given for a general rate law, the order of reactants matters. Hence, we must give the following definition,

Definition 4.2.4 (Affinity network)**.** An *affinity network* $\mathcal{A} \subset \mathcal{S}^* \times [\mathbb{R}^* \to \mathbb{R}]$ is a set of tuples of lists of sites and rate laws.

The interpretation of this is that a tuple $((s_1, \ldots, s_n), f)$ indicates that sites $s_1, \ldots, s_n$ are compatible and may react at rate $f([s_1], \ldots, [s_n])$ where $[s_i]$ denotes the concentration of site s_i. In practice, we will still draw affinity networks as graphs, taking care that the order of arguments is always clear from the context, and adopting the convention that the first argument of a rate law is drawn on the left hand side of a node (that is, in the 9 o'clock position), and subsequent arguments proceed clockwise.

Example 4.2.5 (Enzymes)**.** Consider a system of chemical reactions where an enzyme E binds to a substrate S to form a product P. This can be modelled via mass action kinetics, when we introduce an additional complex C to represent the complex,

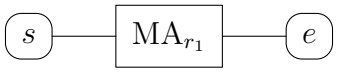
$$S + E \underset{r_{-1}}{\overset{r_1}{\rightleftharpoons}} \mathrm{C} \xrightarrow{r_2} P + E\,.$$

In order to model this system, we must introduce the kinetic law $\mathrm{MA}_r : \mathbb{R}^* \to \mathbb{R}$ defined by $\mathrm{MA}_r([X_1], [X_2], \ldots, [X_n]) = r\,[X_1]\,[X_2] \ldots [X_n]$, corresponding to mass action kinetics

with stoichiometric rate r. Then we may model the system with agents,

$$S \triangleq s(\ell).S^*_\ell \qquad S^*_\ell \triangleq s^*@\ell.E + p^*@l.P$$
$$E \triangleq e(\ell).E^*_\ell \qquad E^*_\ell \triangleq e^*@\ell.E^*_\ell$$

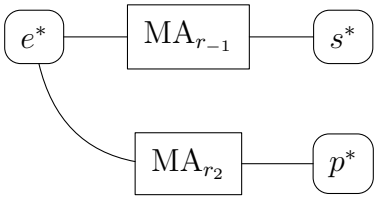
where s, e represent the binding sites of S and E respectively, S^*_ℓ, E^*_ℓ represent bound states of S and E, and s^*, e^*, p^* are *internal sites* which allow interactions between the bound components of a complex. For now we will focus on the first part of the reaction, $S + E \xrightarrow{r_1} C$, whose kinetics can be specified via the affinity network,



This specifies that S and E can react at rate $\mathrm{MA}_{r_1}([S], [E]) = r_1 [E] [S]$ to form the complex,

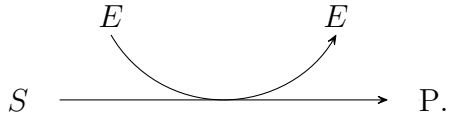
$$C \triangleq (\nu\, \ell)\, (S^*_\ell \mid E^*_\ell)\,.$$

What about the other reactions? These are more difficult to specify as whilst they involve interactions between the virtual site e^* of the bound enzyme with the virtual sites s^* and p^* of the bound substrate. One might assume these could be specified via the affinity network,



however, this is not correct. This that would imply, for example, that the rate of the interaction betweek e^* and p^* would be proportional to the concentration of e^* sites times the concentration of p^* site giving the rate $\mathrm{MA}_{r_2}([p^*], [e^*]) = \mathrm{MA}_{r_2}([C], [C]) = r_2 [C]^2$, however, since only the single pair of e^* and p^* sites which share the same location ℓ corresponding to an individual C molecule are actually able to interact, the actual rate should be $\mathrm{MA}_{r_2}([C]) = r_2 [C]$. In the next section we will see how we can extend our language to properly model mixtures of molecules, and how we may generalise affinity networks to handle internal interactions within a molecule.

Example 4.2.6 (Michaelis-Menten Kinetics)**.** The entirety of the enzyme reaction we considered in the last example can be modelled as a single reaction,



In the case that the enzyme concentration is significantly smaller than the substrate concentration, this may be approximated by the Michaelis-Menten kinetic law,

$$\mathrm{MM}_{V_{\max},K}([S],[E]) \triangleq \frac{V_{\max}[S][E]}{K+[S]}.$$

We are able to model this in continuous-π using agents,

$$S \triangleq s.P \qquad\qquad E \triangleq e.E$$

and affinity network,

Here the order of the arguments most definitely does matter since in general

$$\mathrm{MM}_{V_{\max},K}([S],[E]) \neq \mathrm{MM}_{V_{\max},K}([E],[S]).$$

We can use our convention for drawing affinity networks, to give this network completely explicitly as,

$$\{((s,e),\mathrm{MM}_{V_{\max},K})\}.$$

4.3 A two level calculus for continuous mixtures of species

Real biochemical reactions do not take place between a single molecule of enzyme, and a single molecule of substrate, but rather involve large quantities of each mixed together in certain concentrations. This motivates us to represents solutions of molecules as mixtures of agents, each present at a different real valued concentration. For example, we can represent the mixture of an substrate, enzyme, and product as,

$$[S]\cdot S \parallel [E]\cdot E \parallel [P]\cdot P,$$

specifying that process S is present at concentration $[S]$, process E is present at concentration $[E]$, and process P is present at concentration $[P]$.

We now have two levels of parallel composition: parallel composition of species $|$, and parallel composition of mixtures $\parallel$. In many cases these overlap, for example, we can identify the mixtures $\alpha\cdot(S|E)$ and $\alpha\cdot S \parallel \alpha\cdot E$; we do not wish to distinguish concentration α pairs of S and E molecules, from concentration α of S molecules mixed with concentration α of E molecules. However, we argue that if we consider molecules

as processes, fundamental differences emerge between parallel composition within a single molecule (the species level), and parallel composition across different molecules in a solution (the process level). In our chain of polyethene, each of the CH_2 monomers has two sites bonded together by location s_i, and is bonded to another monomer of CH2 by a location ℓ_i. Now, the links ℓ_i can be used to coordinate the breaking of the chain, and here the location identifies which sites are bound together in a specific way; bound sites at different points along the polymer cannot get together and unbind, since they are not in the same location. In interactions between different polymers, the distinctions between locations does not have the same effect; two monomers which are not only at different locations within the polymer but within entirely different molecules do in fact have the capability to interact just fine, since molecules are floating feely in solution. Therefore, parallel composition of species and of mixtures should not just be two different sides of the same coin, but rather should be treated rather differently. In particular, interactions involving both sites at a location within a molecule, and external to it should be allowed and treated as multiway synchronisations.

Modelling multiway synchronisations across these two levels of composition is a little different than the single level languages we discussed in Chapter 3. For example, a three way interaction may now involve 3 distinct individuals, 3 components of a single individual, or even 2 components of one individual and one of another. The distinction between each of these alternatives is crucial to simulating the system, since interactions within a molecule provide only qualitative information about the molecule's properties as a potential reactant, whereas reactions between molecules are handled quantitatively, and provide the overall rate of the reaction based on the concentration of each reactant. We handle this by extending affinity network to specify for each node a bag of sites we expect to be present within a single agent for a reaction to take place, to give the following revised definition,

Definition 4.3.1 (Affinity network)**.** An *affinity network* $\mathcal{A} \subset (\textsc{Bag}(\mathcal{S}))^* \times [\mathbb{R}^* \to \mathbb{R}]$ is a set of tuples of lists of bags of sites and rate laws.

This allows us to distinguish which sites in a reaction belong to which species, as evinced by Figure 4.1. If we think about the underlying chemical interpretation, we see that these multiway synchronisations are not just some corner case – in fact, any chemical changes internal to a molecule must involve a collision with another particle, even if this is not explicitly stated and the other particle leaves unchanged[3], since some activation

[3]Variants of this scenario give the molecular basis to the role of heat in so called *thermochemical*

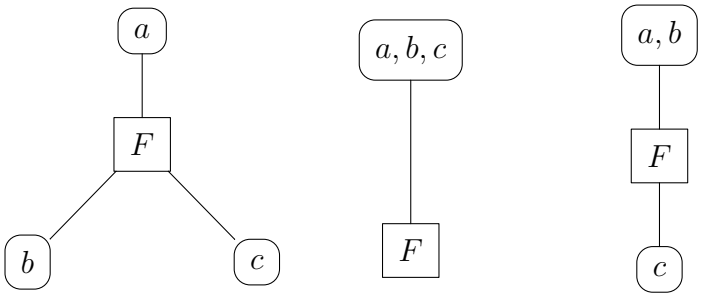


Figure 4.1: *Affinity networks corresponding to sites in three different species, three sites in the same species, and two sites in one species, and one in another.*

energy must be supplied to break and reconfigure the chemical bonds.

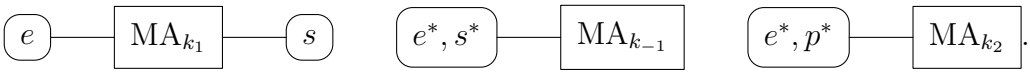
Example 4.3.2 (Enzymes)**.** Example 4.2.5 discussed how we may quantitatively model reactions between substrates and enzymes. However, at this point we were only able to describe the behaviour of the individual agents, and, more troublingly, had no way of specifying the effect of the internal interactions within a molecule. We are now in a position to complete the model using our extended definition of affinity networks. Our complete model still has the same agent definitions as before,

$$S \triangleq s(\ell).S^*_\ell \qquad S^*_\ell \triangleq s^*@\ell.E + p^*@l.P$$
$$E \triangleq e(\ell).E^*_\ell \qquad E^*_\ell \triangleq e^*@\ell.E^*_\ell$$

but we are now able to define represent the whole system as a mixture of agents,

$$\Pi \triangleq [S] \cdot S \parallel [E] \cdot E \parallel [P] \cdot P$$

and give an extended affinity network which includes the internal interactions,



This extended affinity network now includes unary hyperedges which specify that the rate of internal reactions involving sites e^*, s^* and e^*, s^* are $\mathrm{MA}_{r_{-1}}([e^*, s^*]) = r_{-1}\,[C]$ and $\mathrm{MA}_{r_2}([e^*, p^*]) = r_2\,[C]$ respectively.

Example 4.3.3 (Hydrogen Dibromide)**.** The reaction,

$$\mathrm{H_2} \quad + \quad \mathrm{BR_2} \quad \longrightarrow \quad 2\,\mathrm{HBr}$$

reactions.

in which Hydrogen and Bromine are combined to form Hydrogen Dibromide has the interesting kinetic law,

$$\mathrm{L}_k\left(\left[\mathrm{H}_2\right],\left[\mathrm{Br}_2\right],\left[\mathrm{HBr}\right]\right) = \frac{\left[\mathrm{H}_2\right]\left[\mathrm{Br}_2\right]^{1/2}}{1+k\frac{\left[\mathrm{HBr}\right]}{\left[\mathrm{Br}_2\right]}}.$$

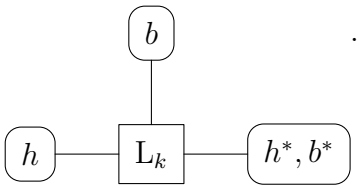
This seems quite far removed from mass action kinetics given the reaction rate depends on the concentration of the product as well as the supposed reactants, and does not have a fixed reaction order, defying the traditional categorisation of reactions as first order reactions (with linear reaction rates), second order reactions (with quadratic rates), etc. It can however, be decomposed into a chain of mass action reactions,

$$\begin{aligned}
\mathrm{Br}_2 &\longrightarrow \mathrm{Br} + \mathrm{Br}\\
\mathrm{Br} + \mathrm{H}_2 &\longrightarrow \mathrm{H} + \mathrm{HBr}\\
\mathrm{H} + \mathrm{Br}_2 &\longrightarrow \mathrm{Br} + \mathrm{HBr}\\
\mathrm{Br} + \mathrm{Br} &\longrightarrow \mathrm{Br}_2.
\end{aligned}$$

This model is, however, unsatisfying, since it replaces a model with one reaction rate parameter, with one with four. We will instead model it as using a general kinetic law, defining the agents[4],

$$\begin{aligned}
\mathrm{H}_2 &\triangleq h(\ell,m).\left(\mathrm{H}^{(\ell)} \mid \mathrm{H}^{(m)}\right)\\
\mathrm{H}^{(\ell)} &\triangleq h^*@\ell.\mathrm{H}^{(\ell)}\\
\mathrm{Br}_2 &\triangleq b(\ell,m).\left(\mathrm{Br}^{(\ell)} \mid \mathrm{Br}^{(m)}\right)\\
\mathrm{Br}^{(\ell)} &\triangleq b^*@\ell.\mathrm{Br}^{(\ell)}
\end{aligned}$$

and the affinity network,



.

This model allows the agents H_2 and Br_2, to dynamically bind to form the complex,

$$(\nu\,\ell,m)\left(\mathrm{H}^{(\ell)} \mid \mathrm{H}^{(m)} \mid \mathrm{Br}^{(\ell)} \mid \mathrm{Br}^{(m)}\right).$$

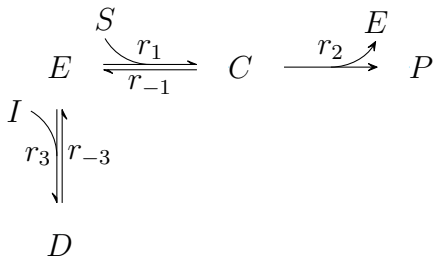
[4]In these agent definitions we write the locations as superscripts rather than subscripts to avoid confusion with subscripts in chemical formulae.

Of course, what we really want is two copies of the complex,

$$\mathrm{HBr} \triangleq (\nu\,\ell)\left(\mathrm{H}^{(\ell)} \mid \mathrm{Br}^{(\ell)}\right)$$

but, as we will see in Section 4.5, the first complex is in fact equivalent to HBr | HBr, so this is the right result after all.

Example 4.3.4 (Competitive inhibition via overlapping sites)**.** Inhibitors are used to alter the rate of reactions via impairing the function of an enzyme. The most common types of of inhibitors are competitive inhibitors which bind to the enzyme in direct competition with the substrate forming a new complex D (as discussed in Sections 2.2 and 2.3.1). These are characterised by the scheme of chemical reactions,



Whilst many competitive inhibitors bind directly to the active site of the enzyme, we will look at the case where the inhibitor binds to another distinct site on the enzyme, and disable the active site via internal interactions (that is, the binding of the inhibitor allosterically modifies the conformity of the active site, so it is no longer compatible with the substrate).

We model these reactions as the process,

$$\Pi \triangleq [S] \cdot S \parallel [I] \cdot I \parallel [E] \cdot E$$

consisting of species representing the substrate S, the inhibitor I, and the the enzyme E.

We use the same substrate definition as before,

$$S \triangleq s(\ell).S^*_\ell \qquad\qquad S^*_\ell \triangleq s^*@\ell.E + p^*@l.P$$

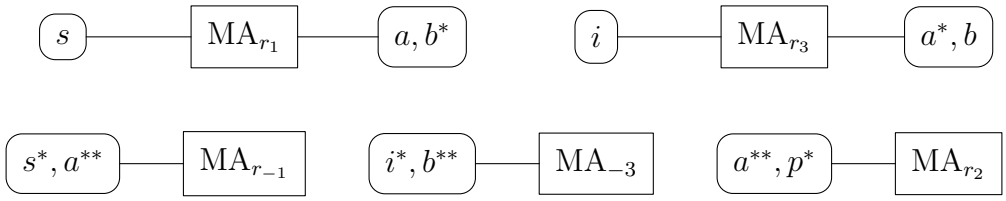
and give a similar definition for the inhibitor,

$$I \triangleq i(\ell).I^*_\ell \qquad\qquad I^*_\ell \triangleq i^*@\ell.I$$

however, we give a new definition for the enzyme,

$$\begin{aligned} E &\triangleq (\nu\,\ell)(A_\ell \mid B_\ell) \\ A_\ell &\triangleq a@\ell(m).A^*_{\ell,m} + a^*@\ell.A_\ell & A^*_{\ell,m} &\triangleq a^{**}@m.A \\ B_\ell &\triangleq b@\ell(m).B^*_{\ell,m} + b^*@\ell.B_\ell & B^*_{\ell,m} &\triangleq b^{**}@m.B \end{aligned}$$

where we define E as the parallel composition of its two reaction sites A_ℓ and B_ℓ. Each site has a bound and unbound state, so for example, the unbound state of the active site is represented by A_ℓ and the bound state is represented by $B^*_{\ell,m}$, where m is the location at which it is bound. For this model we have the affinity network,



This specifies that the enzyme E can undergo one of two reactions: site a of the enzyme can bind to site s of the substrate if site B exposes the virtual site b^* indicating it is in the unbound state, or site b can bind to state i of the inhibitor as long as A exposes virtual site a^* indicating it is unbound. Once bound, the substrate enzyme complex can unbind through internal interactions between s^* and a^{**}, and the inhibitor complex can unbind through internal interactions between i^* and b^{**}. Finally, the bound substrate may produce the product through internal reactions between a^{**} and p^*.

4.4 Syntax and operational semantics

The syntax and semantics of our two level quantitative calculus will build heavily on those we gave for the qualitative bond-calculus in Section 3.3. In fact, the syntax for species S is exactly the same as that of processes in the bond-calculus. On top of these we define *mixtures* or *processes*[5], which consist of solutions of different concentrations of species:

$$P, Q ::= \; c \cdot S \;\mid\; P \parallel Q$$

that is, a mixture is defined recursively as either:

- $c \cdot S$, meaning species S is present in concentration $c \in \mathbb{R}_{\geq 0}$.
- The *parallel composition* or *solution* $P \parallel Q$ of two processes P and Q.

It is now time to move from syntax to semantics, and attempt to link our language with mathematics models which may be simulated. The eventual goal is to derive systems of differential equations, describing the evolution of every mixture of species, however,

[5]These should not be confused with the processes in qualitative calculi, which correspond to species, and are no longer at the top level of the calculus.

we will do this in stages, first considering species in this section, before we go on to consider mixtures of processes in the next. The basic principles for the semantics we give at the species level are still the same as in the qualitative case, with interactions being built up gradually in stages by combining potential transitions and colocating abstractions. However, in our two level calculus, we will not be able to commit to any reactions/synchronisation at the species level, since we have no way of determining the rate until the mixture level, since it will depend in a nonlinear way on both the concentration of the other reactants, and of the current site. Therefore, our transitions will only represent potential reactions, leaving the questions of how these will be committed to the process level.

Example 4.4.1. We will now remind ourselves of how our rules for building up abstractions from Section 3.2 work, by applying them to the enzyme and substrate from Example 4.2.5. In this system, the two halves of the enzyme substrate complex can be describe using two abstractions, one corresponding to the substrate, and another for the enzyme,

$$(\ell)(s^*@\ell.E + p^*@\ell.P) \qquad\qquad (\ell)e^*@\ell.E$$

When we collocate these we get the body of the complex,

$$(\ell)(s^*@\ell.E + p^*@\ell.P) \mid (\ell)(e^*@\ell.E) = (\ell)((s^*@\ell.E + p^*@\ell.P) \mid e^*@\ell.E).$$

When we commit this, we get the species for the complex,

$$\text{commit}((\ell)((s^*@\ell.E + p^*@\ell.P) \mid e^*@\ell.E)) = (\nu\,\ell)((s^*@\ell.E + p^*@\ell.P) \mid e^*@\ell.E))$$

The full transitions rules for species are detailed in Figure 4.2. Unlike for our qualitative calculi, these are used to define a *multitransition system* where transition relation $\rightarrow$ is a bag rather than a set. This is necessary for example, to distinguish between processes $a.\mathbf{0}$ and $a.\mathbf{0} + a.\mathbf{0}$ where the former has only one copy of the a site and the latter has two, since these will yield different reaction rates.

It should be noted that none of these depend on what the affinity network is. This and the fact that we are leaving the question of which potential reactions will be committed to the mixture level lead our transition rules to be much more promiscuous than any calculus we have considered previously – we literally let every site (potentially) interact with any other as long as the locations match up[6]. On the other hand, we do not consider

[6]Of course, when implementing the language one is free to only generate transitions to those whose sites actually appear on the affinity network, however, whilst this significantly improves performance, the end result is the same.

$$\frac{}{\sum_{j=0}^{n} \pi_i.F_i \xrightarrow{\wr \pi_j \int} F_j}\ \textsc{Choice}_{j,n} \qquad \frac{P \equiv_\alpha P' \quad P' \xrightarrow{\alpha} P''}{P \xrightarrow{\alpha} P''}\ \textsc{Alpha}$$

$$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}\ \textsc{Par-Left} \qquad \frac{P \xrightarrow{\alpha} P'}{Q \mid P \xrightarrow{\alpha} Q \mid P'}\ \textsc{Par-Right}$$

$$\frac{P \xrightarrow{\alpha} P' \qquad D(\mathbf{x};\mathbf{l}) \triangleq P}{D(\mathbf{y};\mathbf{m}) \xrightarrow{\alpha} P'\{\mathbf{y}/\mathbf{x},\mathbf{m}/\mathbf{l}\}}\ \textsc{Def} \qquad \frac{P \xrightarrow{\alpha} P' \quad \ell_1,\ldots,\ell_n \notin \text{locations}(\alpha)}{(\nu\, \ell_1,\ldots,\ell_n)P \xrightarrow{\alpha} (\nu\, \ell_1,\ldots,\ell_n)P'}\ \textsc{Res}$$

$$\frac{P \xrightarrow{\alpha} P'}{(\nu\, \ell_1,\ldots,\ell_n)P \xrightarrow{\text{delocate}(\alpha)} (\nu\, \ell_1,\ldots,\ell_n)P'}\ \textsc{Delocate}$$

$$\frac{P \xrightarrow{\alpha} P' \quad Q \xrightarrow{\beta} Q' \quad \exists \ell \neq \top,\ \text{locations}(\alpha) = \text{locations}(\beta) = \{\ell\}}{P \mid Q \xrightarrow{\alpha \uplus \beta} P' \mid Q'}\ \textsc{Com}$$

Figure 4.2: *Species multitransition system rules.*

any reactions between unlocated sites, leaving these to be handled at the mixture level. This is because, even if two unlocated sites of a species are compatible, if they are on the same molecule they cannot interact unless they are at the same location on the molecule – counterintuitively sites on different molecules can often interact more easily than sites on the same molecule since the molecules are suspended in a well mixed solution, and can moving around freely, bring any location on one molecule into contact with any location on another, whilst, on the other hand, two sites on the same molecule cannot bump into each other[7]. Also, when interactions at shared locations reach the mixture level, they will lose their location information, becoming indistinguishable from unlocated sites. This will be done using the delocate function,

Definition 4.4.2. We define the *delocation* of a bag of located sites α by,

$$\begin{aligned} \text{delocate}(\alpha \uplus \beta) &= \text{delocate}(\alpha) \uplus \text{delocate}(\beta) \\ \text{delocate}(\wr a @ \ell \int) &= \wr a \int \\ \text{delocate}(\varnothing) &= \varnothing. \end{aligned}$$

We will not propagate the location information to the mixture level, since the location of a site on a particular molecule cannot prevent another molecule in solution reacting

[7]That is, assuming the molecule is rigid. Whilst this assumption holds for point particle like molecules, it may not be realistic for long polymers such as DNA.

with it.

Now to look at some of the rules in detail. The rule COM combines transitions, taking the bag union of their sites, based on the rule that everything (potentially) interacts with everything else, as long as they share a location; it is only at the process level that we will use the affinity network to filter down to those reactions which occur at an appreciable rate. The combined effect of the RES, PAR-LEFT, and PAR-RIGHT rules is to allow potential communications to build and bubble up to the top. However, at some point they might meet a location restriction clause $(\nu\,\ell)$, which sites at that location cannot pass. So how do these transitions reach the mixture level? At any moment[8] a bag of located sites $\wr a_1@\ell, \ldots, a_n@\ell\wr$ can be delocated, turning into into a bag of ambient sites $\wr a_1, \ldots, a_n\wr$, preventing it from being built into any new interactions with COM, but removing location information and allowing it to pass through all restrictions, and potentially, allow other sites from other molecules in the synchronisation. This is done using the DELOCATE rule.

4.5 Structural congruence, normal forms, and prime species

We have now defined our transitions system for species in reference to the syntactic description of agents, with the exception for the rule ALPHA, which allows for α-equivalent agents to be exchanged freely. This directness aids in implementing and reasoning about the language, since in order to know what transitions we can take starting from a given agent, we only need to handle α-equivalence[9]. This does however leave us with three main problems:

1. The multitransition system will be much larger than necessary, since it will includes transitions from many equivalent.

2. When simulating the system and calculating quantitative rates, we need to be able to identify which species are the same.

3. After reaction we need to put the products in a form where it is clear what distinct molecules it contains.

[8] Actually, we only let this happen when it is needed to let a transition pass through a restriction, in order to avoid counting these transitions twice: once when their delocated version bubbles up, and located version is bubbled up and subsequently delocated.

[9] This is notably simpler than for the π-calculus and continuous π. We have internal mobility to thank for this, since Davide Sangiorgi showed that internal mobility allows the transition system π-calculus to be reformulated using only α-congruence, rather than structural congruence [105].

The first of these points is a major practical concern when implementing the system, since the transition systems of many finite dimensional systems become infinite dimensional if we do not identify equivalent agents. For example, in the enzyme example, the products of the breakdown of the enzyme substrate complex are $E \mid S$ but if we apply rules blindly we may find them to be $(\nu\,\ell_1)(E|S) \equiv (\nu\,\ell_1, \ell_2)(E|S) \equiv \ldots$; whilst this example is easy to avoid, unless we have a systematic way of identifying equivalent species, we will soon end up infinitely inflating the transition systems of many reasonable models. This example also illustrates the other two problems, for how can we keep track of the populations of even simple species such as S if they may be present in the system in a different form as a duplicated species, or trapped within a false complex such as $(\nu\,\ell_1)(E \mid S)$?

To deal with these problems, we are motivated to identify equivalent processes at the syntactic level by defining a structural congruence relation $\equiv$ which identifies equivalent species, mixtures, and abstractions. Then we will be able to think in terms of equivalence classes of agents upto structural congruence, removing the duplication in the representation of processes. This structural congruence is given by in the next three definitions.

Definition 4.5.1. The structural congruence $\equiv$ on species is the least congruence (equivalence relation, preserved all operations) containing α-equivalence and satisfying the following axioms:

$$\begin{aligned}
\mathbf{0} \mid A &\equiv A \\
A \mid B &\equiv B \mid A \\
(A \mid B) \mid C &\equiv A \mid (B \mid C) \\
\textstyle\sum_{i=0}^{n} \pi_i.A_i &\equiv \textstyle\sum_{i=0}^{n} \pi_{\sigma_i}.A_{\sigma_i} && \text{given } \sigma \text{ perm.} \\
(\nu\,\ell)F &\equiv F && \text{given } \ell \notin \text{flocs}(F) \\
(\nu\,\ell)(A \mid B) &\equiv A \mid (\nu\,\ell)B && \text{given } \ell \notin \text{flocs}(F)
\end{aligned}$$

Definition 4.5.2. The structural congruence $\equiv$ on abstractions is the least congruence containing α-equivalence and satisfying the following axioms:

$$\begin{aligned}
(\ell_1, \ldots, \ell_{n-1}, \ell_n)A &\equiv (\ell_1, \ldots, \ell_{n-1})A && \text{given } \ell_n \notin \text{flocs}(A) \\
(\ell_1, \ldots, \ell_n)A &\equiv (\ell_1, \ldots, \ell_n)B && \text{given } A \equiv B
\end{aligned}$$

Definition 4.5.3. The structural congruence $\equiv$ on processes is the least congruence

containing α-equivalence and satisfying the following axioms:

$$\begin{aligned}
(c \cdot \mathbf{0}) \parallel P &\equiv P \\
P \parallel Q &\equiv Q \parallel P \\
(P \parallel Q) \parallel R &\equiv P \parallel (Q \parallel R) \\
(c + d) \cdot A &\equiv (c \cdot A) \parallel (d \cdot A) \\
c \cdot A &\equiv c \cdot B \qquad\qquad \text{given } A \equiv B
\end{aligned}$$

Now we have a definition of syntactic equivalence of agents, however, as easy as working with equivalence classes of agents may be for a theoretician, it is less obvious how an implementation of the language may do this, given at some level some concrete syntactic representation must be given to the agents of the system. In fact, this will not be too difficult, given the following result which shows that structural congruence is a bisimulation of agents, which in particular implies that if we simply pick one representative element from each equivalence class, this will give the correct transition system. That is, in so far as the transition semantics is concerned, any choice of representatives from the equivalence classes is sufficent to give the behaviour of the whole class.

Proposition 4.5.4. *Structural congruence on species is a strong bisimilarity of agents. That is, given any A, B such that $A \equiv B$ we have that,*

$$A \xrightarrow{\alpha} A' \qquad \Rightarrow \qquad \exists B' \equiv A',\ \ B \xrightarrow{\alpha} B' \ .$$

Proof. We can check this holds for each of this structural congruence rules and transition rules. The result then follows by induction on the structure of derivations. □

Now we are reassured that we lose nothing by picking a single element to represent each equivalence class, we are left with the problem of how to pick this element. We will do this by defining a *normal form* for species, which gives a consistient format for agents, breaking all of the symmetries induced by structural congruence so that two agents are equivalent iff they have the same normal form. This means we have now resolved the second problem, since we will be able to tell agents are equivalent by comparing their normal form.

Definition 4.5.5. The normal form for processes, species, and abstractions are defined

via the following grammer,

$$\begin{aligned}
\textsc{Proc} &::= \|_{i=1}^{n}\, \alpha_i \cdot \textsc{Spec} & \textsc{Spec} &::= \,|_{i=1}^{n} \textsc{Res} \\
\textsc{Res} &::= \,(\nu\, \ell_1, \ldots, \ell_n)\textsc{Par} & \textsc{Par} &::= \,|_{i=1}^{n} \textsc{Sum} \\
\textsc{Sum} &::= \,\textstyle\sum_{i=0}^{n} \pi.\textsc{Abst} & \textsc{Abst} &::= \,(\ell_1, \ldots, \ell_n)\textsc{Spec},
\end{aligned}$$

on which we impose the following conditions:

- The order of the terms in a Proc, Spec, Par, or Sum, and the order of locations in a Res do not matter (we can either apply some canonical ordering on these, or store them in an order-independent data structure).
- The names of bound location in Res or Abst do not matter (we can choose canonical names using an extension of De Bruijn indicies, or use any of the other standard techniques for handling α-conversion).
- We assume Proc contains no duplicate species.
- We assume Res contains no redundant locations, and that the last location of Abst is not redundant.
- We assume that there is no partition of locations in a Res which would allow it to be split into a parallel composition of two restrictions.

The following results assure us that we normal forms act as the unique representatives of the equivalence classes under structural congruence.

Proposition 4.5.6. *Every process (species, abstractions) P has a unique normal form* $\mathrm{nf}(P)$ *to which it is structurally congruent.*

Proposition 4.5.7. *Let P, Q be processes (species, abstractions). Then,*

$$P \equiv Q \qquad \Leftrightarrow \qquad \mathrm{nf}(P) = \mathrm{nf}(Q).$$

Finally, we address the third problem, of identifying the distinct molecules in a species. Intuitively, we know a species can be considered as two separate molecules if is the parallel composition of two completely independent molecules. Whilst is it obvious that species such as $E \mid S$ can be split up, it less obvious why more complex species such as $(\nu\, \ell, m)(A_\ell \mid B_m)$ can be split up whilst others such as $(\nu\, \ell, m)(A_\ell \mid B_m \mid C_{\ell,m})$ cannot. We address this by defining the concept of a *prime species* based on our structural congruence relation.

Definition 4.5.8 (Prime species)**.** A species $S \not\equiv \mathbf{0}$ is prime if we have that,

$$S \equiv A \mid B \qquad \Rightarrow \qquad A \equiv \mathbf{0} \text{ or } B \equiv \mathbf{0}$$

for all species A, B.

We now define the prime decomposition of S as the unique decomposition of S into a parallel composition of prime species,

Definition 4.5.9. We define the prime decomposition $\text{primes}(S)$ of species S as the bag of prime species $\wr S_1, \ldots, S_n \int$ such that,

$$S \equiv S_1 \mid \ldots \mid S_n.$$

In order for this to be well defined we need the following result,

Proposition 4.5.10. *The prime decomposition* $\text{primes}(S)$ *of any species* S *exists, and is unique up to structural congruence.*

In fact, the way we have defined the normal form gives as a very direct way of computing the prime decomposition of S, thanks to the final condition of Definition 4.5.5 which ensures that normal form of S will be a parallel composition of prime species.

Lemma 4.5.11. *If a species* S *has normal form* $\text{nf}(S) = S_1 \mid \ldots \mid S_n$*, then it has prime decomposition* $\text{primes}(S) = \wr S_1, \ldots, S_n \int$.

This section culminates in the definition of the class $\mathcal{P}$ of prime species. As we reach the end of our treatment of species, this will give us ideal version of species to base our process semantics on: correspond to independent molecules and are distinct up to structural congruence.

Definition 4.5.12. We define $\mathcal{P}$ as the class of equivalence classes of prime species under structural congruence. Alternatively, $\mathcal{P}$ can be identified with the class of prime normal forms,

$$\mathcal{P} = \Big\{ \text{nf}(P) \ : \ P \text{ prime} \Big\}.$$

Example 4.5.13 (Hydrogen Dibromide)**.** In Example 4.3.3, we saw our reactants H_2 and Br_2 bind together, to form the complex,

$$C \triangleq (\nu\, \ell, m) \left(\mathrm{H}^{(\ell)} \mid \mathrm{H}^{(m)} \mid \mathrm{Br}^{(\ell)} \mid \mathrm{Br}^{(m)} \right),$$

which as not quite what we wanted, as it tangles together the two copies of, $\mathrm{HBr} \triangleq (\nu\,\ell)\left(\mathrm{H}^{(l)} \mid \mathrm{Br}^{(l)}\right)$, which the reaction is expected to produce. This problem is, however, resolved when we put the result in normal form as,

$$\mathrm{nf}(C) = \mathrm{HBr} \mid \mathrm{HBr}.$$

Example 4.5.14. In Example 3.3.14 we derived the products of the reactions

$$A + B + E \longrightarrow AB + E$$
$$A + B + C + D + E \longrightarrow AB + CD + E$$

as the restrictions,

$$P \triangleq (\nu\,\ell)(A_\ell \mid B_\ell \mid E) \qquad\qquad Q \triangleq (\nu\,\ell, m)(A_\ell \mid B_\ell \mid C_m \mid D_m \mid E),$$

respectively. These can be put into normal form,

$$\mathrm{nf}(P) = (\nu\,\ell)(A_\ell \mid B_\ell) \mid E$$
$$\mathrm{nf}(Q) = (\nu\,\ell)(A_\ell \mid B_\ell) \mid (\nu\,\ell)(C_\ell \mid D_\ell) \mid E$$

revealing that the first consists of the two prime species/independent molecules $AB \triangleq (\nu\,\ell)(A_\ell \mid B_\ell)$ and E, whilst the second consists of three molecules $AB \triangleq (\nu\,\ell)(A_\ell \mid B_\ell)$, $CD \triangleq (\nu\,\ell)(C_\ell \mid D_\ell)$, and E.

4.6 Process level vector field semantics

We are now finally ready to build upon the non-deterministic transition semantics we have defined for species, and define a vector field semantics for process. This is based on the *process space*, a vector space comprising of tensors between unlabelled transitions and bag of sites. This will allow us to encode a prime species S as a vector $|S\rangle$, describing of all of its potential transitions, and encode processes as linear combinations of prime species.

Definition 4.6.1. The *process space* is defined as the tensor space $\mathbb{P} = (\mathcal{P} \times \textsc{Abst}) \otimes \textsc{Bag}(\mathcal{S})$. That is, the process space is the vector space generated by the basis of pure tensors,

$$|S \to S'\rangle|a\rangle \triangleq (S, S') \otimes a$$

for a prime species $S \in \mathcal{P}$, an abstraction $S' \in \textsc{Abst}$, and a bag of sites $a \in \textsc{Bag}(\mathcal{S})$.

We will use the Dirac's bra-ket notation as a shorthand for vectors in this tensor space. So, for example, we can write $(S, S') \otimes a$ as $|S \to S'\rangle|a\rangle$. This notation is convenient as it lets us easily work with impure tensors such as $(\alpha|A \to A'\rangle + \beta|B \to B'\rangle)|s\rangle$ and $|S \to S'\rangle(\alpha|a\rangle + \beta|b\rangle)$.

We also define the species vectors, corresponding to the prime species – this forms an orthogonal basis for a subspace of $\mathbb{P}$ corresponding to mixtures of species.

$$|S\rangle \triangleq \sum\Big\{|S \to S'\rangle|\beta\rangle : S \xrightarrow{\alpha} S' \text{ where } \beta = \text{delocate}(\alpha)\Big\}$$

The space $\mathbb{P}$ suffices to describe all possible mixtures, for any affinity network. If we are working with respect to a specific affinity/compatibility network $\mathcal{M}$, we can restrict ourself to the subspace $\mathbb{P}_{\mathcal{M}}$, which is the image of $\mathbb{P}$ under the projection defined by,

$$\text{Proj}_{\mathcal{M}}\,|A \to A'\rangle|\alpha\rangle = \begin{cases} |A \to A'\rangle|\alpha\rangle & \text{if } \alpha \in \text{nodes}(\mathcal{M}) \\ \mathbf{0} & \text{otherwise} \end{cases}$$

This projection also gives a corresponding species basis for $\mathbb{P}_{\mathcal{M}}$,

$$\text{Proj}_{\mathcal{M}}\,|S\rangle = \sum\Big\{|S \to S'\rangle|\beta\rangle : S \xrightarrow{\alpha} S' \text{ where } \beta = \text{delocate}(\alpha) \text{ and } \beta \in \text{nodes}(\mathcal{M})\Big\}$$

The definitions we give throughout this section will work equally well for the full process space $\mathbb{P}$, and its subspaces $\mathbb{P}_{\mathcal{M}}$, and we will use $|S\rangle$ to represent the elements of the appropriate species bases interchangeably.

We then use this to define the species embedding by linearity,

$$|S\rangle \triangleq \sum_{P \in \text{primes}(S)} |P\rangle$$

We also define observables, which let us measure the concentration of a given state. We define bras by,

$$\langle S_j \to S_j'|\langle s_j|\left(\sum_i \alpha_i|S_i \to S_i'\rangle|s_i\rangle\right) = \alpha_j;$$

this means a pure bra $\langle\Phi|$ is the projection from vectors to their components for the corresponding ket $|\Phi\rangle$.

We do, however, have the problem that if we want to project onto more general vectors, they may not have unit length, so the previous definition would project in the right direction, but scale the result. Hence, we are motivated to define the concentration of a vector $|\Phi\rangle$ as,

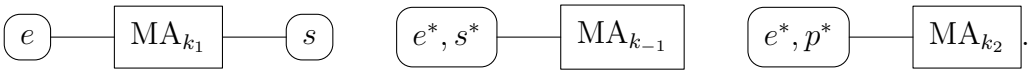
$$\langle\widehat{\Phi}| = \langle\Phi|/\langle\Phi|\Phi\rangle$$

Now we can use this to measure the concentration of a single species in a mixture.

Example 4.6.2. In our enzyme example, we had species,

$$S \triangleq s(\ell).S^*_\ell \qquad S^*_\ell \triangleq s^*@\ell.E + p^*@l.P$$
$$E \triangleq e(\ell).E^*_\ell \qquad E^*_\ell \triangleq e^*@\ell.E^*_\ell$$
$$P \triangleq p.P$$

affinity network,



and we have process definition (now explicitly including complex $C \triangleq (\nu\,\ell)(S_\ell \mid E_\ell)$),

$$\Pi \triangleq [S] \cdot S \parallel [E] \cdot E \parallel [P] \cdot P \parallel [C] \cdot C.$$

We can represent this as a process vector in $\mathbb{P}_{\mathcal{M}}$,

$$\begin{aligned}
|\Pi\rangle &= [S]\,|S\rangle + [E]\,|E\rangle + [P]\,|P\rangle + [C]\,|C\rangle \\
&= [S]\,|S \to (\ell)S^*_\ell\rangle|s\rangle + [E]\,|E \to (\ell)E^*_\ell\rangle|e\rangle + [P]\,|P \to P\rangle|p\rangle \\
&+ [C]\,(|C \to S|E\rangle|s^*, e^*\rangle + |C \to P|E\rangle|p^*, e^*\rangle) \\
&= [S]\,|S \to (\ell)S^*_\ell\rangle|s\rangle + [E]\,|E \to (\ell)E^*_\ell\rangle|e\rangle + [P]\,|P \to P\rangle|p\rangle \\
&+ [C]\,|C \to S|E\rangle|s^*, e^*\rangle + [C]\,|C \to P|E\rangle|p^*, e^*\rangle.
\end{aligned}$$

Suppose we want to extract the C component of this vector. We can do this using the normalized bra $\langle\widehat{C}|$ to project onto the corresponding subspace,

$$\begin{aligned}
\langle\widehat{C}|\Pi\rangle &= \frac{1}{2}(\langle C \to S|E|\langle s^*| + \langle C \to P|E|\langle p^*|)|\Pi\rangle \\
&= \frac{1}{2}([C]\,(\langle C \to S|E|\langle s^*, e^*|)(|C \to P|E\rangle|p^*, e^*\rangle) \\
&\qquad + [C]\,(\langle C \to S|E|\langle s^*, e^*|)(|C \to P|E\rangle|p^*, e^*\rangle)) \\
&= \frac{1}{2}([C] + [C]) \\
&= [C]
\end{aligned}$$

In this way we are able to find the concentration of any species in $|\Pi\rangle$.

The process space $\mathbb{P}_{\mathcal{M}}$ give the configuration space of possible mixtures of species with respect to an affinity network $\mathcal{M}$, whilst process vectors $|\Pi\rangle$ represent mixtures of species within it. However, in order to properly understand a biological or chemical system, we need to understand how it evolves over time. We do this by defining the time

evolution vector $\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t}$, which gives the state of the system after an infinitesimal amount of time has elapsed. But first, we need two auxiliary definitions, for the *interaction concentration of a process* $|\Pi\rangle$ *at site* s, $\mathrm{conc}_s|\Pi\rangle$, and the *interaction direction of a process* $|\Pi\rangle$ *at site* s, $\mathrm{direct}_s|\Pi\rangle$; for a given site s, these measure respectively, the total concentration of this site in a process, and the averaged direction of evolution interaction at this site would cause. There is a parallel between these definitions and the concept of partial measurement in Quantum Mechanics: when you measure the direction of P at site s, you don't just have a single transition telling you what direction to go in, but a superposition of transitions, telling you the probability/concentration of each transition which you could take.

Definition 4.6.3 (Interaction concentration)**.** The *interaction concentration of a process* $P \in P$ *at site* s, is defined by

$$\mathrm{conc}_s|\Pi\rangle = \|(\mathbb{I}\otimes\langle s|)|\Pi\rangle\|_1\,,$$

where $\mathbb{I}$ is the unit operator on the transition space (so $\mathbb{I}|S\to S'\rangle \triangleq |S\to S'\rangle$), and $\|\cdot\|_1$ is the ℓ^1-norm, $\|\sum_i \alpha_i|S\to S'\rangle\|_1 \triangleq \sum_i|\alpha_i|$. Explicitly, this means,

$$\mathrm{conc}_s\left(\sum_m\left(\sum_i \alpha_{i,m}|S_{i,m}\to S'_{i,m}\rangle\right)|m\rangle\right) = \sum_i|\alpha_{s,i}|\,.$$

Definition 4.6.4 (Interaction direction)**.** The *interaction direction of a process* $|\Pi\rangle$ *at site* s is defined as,

$$\mathrm{direct}_s|\Pi\rangle = \begin{cases}(\mathbb{I}\otimes\langle s|)|\Pi\rangle/\,\mathrm{conc}_s|\Pi\rangle & \text{if } \mathrm{conc}_s|\Pi\rangle\neq 0\\ \mathbf{0} & \text{otherwise}\end{cases}.$$

Explicitly, this means,

$$\mathrm{direct}_s\left(\sum_m\left(\sum_i \alpha_{i,m}|S_{i,m}\to S'_{i,m}\rangle\right)|m\rangle\right) = \sum_i\left(\frac{\alpha_{i,s}}{\sum_j|\alpha_{s,j}|}\right)|S_{i,s}\to S'_{i,m}\rangle.$$

In fact, we can use these definitions to decompose any interaction vector, first by sites, and then into concentrations and directions at each site,

Proposition 4.6.5. *For any vector* $|\Pi\rangle$*, we have the following decomposition,*

$$|\Pi\rangle = \sum_s(\mathrm{conc}_s|\Pi\rangle)(\mathrm{direct}_s|\Pi\rangle)|s\rangle$$

Proof. By the definitions of conc and direct, for any s we have that,

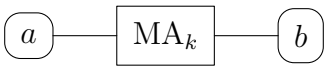
$$(\mathrm{conc}_s\,|\Pi\rangle)(\mathrm{direct}_s\,|\Pi\rangle) = (\mathbb{I}\otimes\langle s|)|\Pi\rangle.$$

But this is just the projection of $|\Pi\rangle$ onto the subspace $\mathbb{P}_s$ spanned by the vectors $|S\to S'\rangle|s\rangle$ for all S, S'. The subspaces $\mathbb{P}_s$ are orthogonal and $\mathbb{P} = \bigoplus_s \mathbb{P}_s$ so this decomposition holds. □

Example 4.6.6. Consider the system with agents,

$$A \triangleq a.A' + b.A'' \qquad\qquad B \triangleq b.B'$$

affinity network,



and the mixture,

$$\Pi \triangleq \alpha\cdot A \parallel \beta\cdot B.$$

Then, we have

$$\begin{aligned}|\Pi\rangle &= \alpha|A\rangle + \beta|B\rangle\\ &= \alpha|A\to A'\rangle|a\rangle + \alpha|A\to A''\rangle|b\rangle + \beta|B\to B'\rangle|b\rangle.\end{aligned}$$

Measuring the concentration and direction of $|\Pi\rangle$ at $|a\rangle$ is straightforward, as it corresponds to a single transition,

$$\begin{aligned}\mathrm{conc}_a\,|\Pi\rangle &= \|(\mathbb{I}\otimes\langle a|)(\alpha|A\to A'\rangle|a\rangle + \alpha|A\to A''\rangle|b\rangle + \beta|B\to B'\rangle|b\rangle)\|_1\\ &= \|\alpha|A\to A'\rangle\langle a|a\rangle + \alpha|A\to A''\rangle\langle a|b\rangle + \beta|B\to B'\rangle\langle a|b\rangle\|_1\\ &= \|\alpha|A\to A'\rangle\|_1\\ &= |\alpha|\\ \mathrm{direct}_a\,|\Pi\rangle &= \frac{1}{|\alpha|}(\mathbb{I}\otimes\langle a|)|\Pi\rangle\\ &= \frac{1}{|\alpha|}\alpha|A\to A'\rangle\\ &= (\mathrm{sign}\,\alpha)|A\to A'\rangle\end{aligned}$$

where $\mathrm{sign}\,\alpha = 1$ if $\alpha > 0$ and 0 otherwise.

Measuring the concentration and direction of $|\Pi\rangle$ at $|b\rangle$ is more difficult, given multiple processes offer the same site; this means if we see the site b, we do not know which species

it belongs to. However, the definitions we have given for conc and direct handle this case by summing over the possibilities, allowing us to perform the following calculations,

$$\begin{aligned}
\mathrm{conc}_b\,|\Pi\rangle &= \|(\mathbb{I}\otimes\langle b|)(\alpha|A\to A'\rangle|a\rangle + \alpha|A\to A''\rangle|b\rangle + \beta|B\to B'\rangle|b\rangle)\|_1\\
&= \|\alpha|A\to A'\rangle\langle b|a\rangle + \alpha|A\to A''\rangle\langle b|b\rangle + \beta|B\to B'\rangle\langle b|b\rangle\|_1\\
&= \|\alpha|A\to A''\rangle + \beta|B\to B''\rangle\|_1\\
&= |\alpha| + |\beta|\\
\mathrm{direct}_b\,|\Pi\rangle &= \frac{1}{|\alpha|+|\beta|}(\mathbb{I}\otimes\langle b|)|\Pi\rangle\\
&= \frac{\alpha}{|\alpha|+|\beta|}|A\to A''\rangle + \frac{\beta}{|\alpha|+|\beta|}|B\to B'\rangle.
\end{aligned}$$

This means that the total concentration of site b across all species of the mixture is $|\alpha|+|\beta|$, and that a proportion $\frac{\alpha}{|\alpha|+|\beta|}$ of these sites are associated with the transition $A\to A''$, whilst a proportion $\frac{\beta}{|\alpha|+|\beta|}$ of these sites are associated with the transition $B\to B'$.

Together this gives us the site decomposition,

$$\begin{aligned}
|\Pi\rangle = {}& |\alpha|\,(\mathrm{sign}(\alpha)|A\to A'\rangle)|a\rangle\\
&+ (|\alpha|+|\beta|)\left(\frac{\alpha}{|\alpha|+|\beta|}|A\to A''\rangle + \frac{\beta}{|\alpha|+|\beta|}|B\to B'\rangle\right)|b\rangle.
\end{aligned}$$

We are now ready to start defining the effect of a given reaction. For this we first define the *reaction vector* $\mathrm{React}(|T_1\rangle,\ldots,|T_n\rangle)$ resulting from a reaction involving transition vectors $|T_1\rangle,\ldots,|T_n\rangle$.

Definition 4.6.7. The *reaction vector* $\mathrm{React}(|T_1\rangle,\ldots,|T_n\rangle)$ is defined on unit transactions $|T_1\rangle = |S_1\to S_1'\rangle,\ldots,|S_n\to S_n'\rangle$ as follows,

$$\mathrm{React}(|T_1\rangle,\ldots,|T_n\rangle) \triangleq |\,\mathrm{commit}(S_1',\ldots,S_n')\rangle - |S_1\rangle - \ldots - |S_n\rangle.$$

This definition is then extended by multi-linearity to arbitrary transition vectors; that is we set,

$$\mathrm{React}\left(|T_1\rangle,\ldots,|T_i\rangle = \textstyle\sum_j \alpha_{i,j}|T_{i,j}\rangle,\ldots,|T_n\rangle\right) = \sum_j \alpha_{i,j}\,\mathrm{React}(|T_1\rangle,\ldots,|T_{i,j}\rangle,\ldots,|T_n\rangle).$$

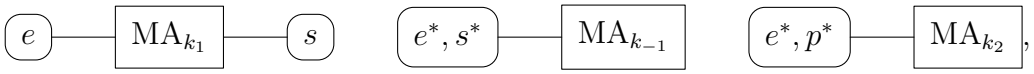
Finally, we may define the *evolution vector* $\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t}\in\mathbb{P}_{\mathcal{M}}$ of a mixture $|\Pi\rangle\in\mathbb{P}_{\mathcal{M}}$, which captures the instantaneous velocity of evolution of a system starting from the given mixture.

Definition 4.6.8. We define the *instantaneous evolution vector* of a vector $|\Pi\rangle \in \mathbb{P}_{\mathcal{M}}$ with respect to affinity network $\mathcal{M}$ as,

$$\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t} \triangleq \sum_{L(a_1,\ldots,a_n)\in\mathcal{M}} L\left(\mathrm{conc}_{a_1}|\Pi\rangle,\ldots,\mathrm{conc}_{a_n}|\Pi\rangle\right)\mathrm{React}(\mathrm{direct}_{a_1}|\Pi\rangle,\ldots,\mathrm{direct}_{a_n}|\Pi\rangle).$$

Example 4.6.9. Recall that in the enzyme example, we have,

$$\Pi \triangleq [S]\cdot S \parallel [E]\cdot E \parallel [P]\cdot P \parallel [C]\cdot C,$$



and,

$$\begin{aligned}|\Pi\rangle = [S]\,|S \to (\ell)S^*_\ell\rangle|s\rangle + [E]\,|E \to (\ell)E^*_\ell\rangle|e\rangle + [P]\,|P \to P\rangle|p\rangle \\ + [C]\,|C \to S|E\rangle|s^*, e^*\rangle + [C]\,|C \to P|E\rangle|p^*, e^*\rangle.\end{aligned}$$

We can then calculate the evolution vector as,

$$\begin{aligned}\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t} &= \mathrm{MA}_{k_1}(\mathrm{conc}_e|\Pi\rangle, \mathrm{conc}_s|\Pi\rangle)\,\mathrm{React}\left(\mathrm{direct}_e|\Pi\rangle, \mathrm{direct}_s|\Pi\rangle\right) \\ &\quad + \mathrm{MA}_{k_2}(\mathrm{conc}_{e^*,s^*}|\Pi\rangle)\,\mathrm{React}\left(\mathrm{direct}_{e^*,s^*}|\Pi\rangle\right) \\ &\quad + \mathrm{MA}_{k_3}(\mathrm{conc}_{e^*,p^*}|\Pi\rangle)\,\mathrm{React}\left(\mathrm{direct}_{p^*,s^*}|\Pi\rangle\right) \\ &= \mathrm{MA}_{k_1}([E],[S])\,\mathrm{React}\left(|E \to (\ell)E^*_\ell\rangle, |S \to (\ell)S^*_\ell\rangle\right) \\ &\quad + \mathrm{MA}_{k_2}([C])\,\mathrm{React}\left(|C \to E|S\rangle\right) \\ &\quad + \mathrm{MA}_{k_3}([C])\,\mathrm{React}\left(|C \to E|P\rangle\right) \\ &= k_1\,[E]\,[S]\,(|C\rangle - |E\rangle - |S\rangle) \\ &\quad + k_2\,[C]\,(|E\rangle + |S\rangle - |C\rangle) \\ &\quad + k_3\,[C]\,(|E\rangle + |P\rangle - |C\rangle).\end{aligned}$$

From this, we can see that this corresponds to the system of ODEs,

$$\begin{aligned}\frac{\mathrm{d}\,[S]}{dt} &= -k_1\,[E]\,[S] + k_2\,[C] \\ \frac{\mathrm{d}\,[E]}{dt} &= -k_1\,[E]\,[S] + (k_2+k_3)\,[C] \\ \frac{\mathrm{d}\,[C]}{dt} &= k_1\,[E]\,[S] - (k_2+k_3)\,[C] \\ \frac{\mathrm{d}\,[P]}{dt} &= k_3\,[C]\,.\end{aligned}$$

Example 4.6.10. Recall Example 4.6.6 where we had,

$$\Pi \triangleq \alpha \cdot A \parallel \beta \cdot B,$$

and,

We can then calculate the evolution vector as,

$$\begin{aligned}
\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t} &= \mathrm{MA}_k(\mathrm{conc}_a\,|\Pi\rangle, \mathrm{conc}_b\,|\Pi\rangle)\,\mathrm{React}\,(\mathrm{direct}_a\,|\Pi\rangle, \mathrm{direct}_b\,|\Pi\rangle) \\
&= \mathrm{MA}_k([A]\,,[A]+[B])\,\mathrm{React}\left(|A\to A'\rangle, \frac{[A]}{[A]+[B]}|A\to A''\rangle + \frac{[B]}{[A]+[B]}|B\to B''\rangle\right) \\
&= k\,[A]\,([A]+[B])\left(\frac{[A]}{[A]+[B]}(|A'\rangle + |A''\rangle - 2|A\rangle)\right. \\
&\qquad\qquad \left. + \frac{[B]}{[A]+[B]}(|A'\rangle + |B'\rangle - |A\rangle - |B\rangle)\right) \\
&= k\,[A]^2\,(|A'\rangle + |A''\rangle - 2|A\rangle) + k\,[A]\,[B]\,(|A'\rangle + |B'\rangle - |A\rangle - |B\rangle).
\end{aligned}$$

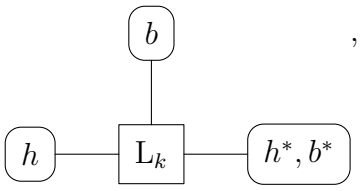
Example 4.6.11 (Hydrogen Dibromide)**.** In Example 4.3.3 we modelled the reaction,

$$\mathrm{H}_2 \quad + \quad \mathrm{BR}_2 \quad \longrightarrow \quad 2\,\mathrm{HBr}$$

via the agents,

$$\begin{aligned}
\mathrm{H}_2 &\triangleq h(\ell, m).\left(\mathrm{H}^{(\ell)} \mid \mathrm{H}^{(m)}\right) \\
\mathrm{H}^{(\ell)} &\triangleq h^*@\ell.\mathrm{H}^{(\ell)} \\
\mathrm{Br}_2 &\triangleq h(\ell, m).\left(\mathrm{Br}^{(\ell)} \mid \mathrm{Br}^{(m)}\right) \\
\mathrm{Br}^{(\ell)} &\triangleq b^*@\ell.\mathrm{Br}^{(\ell)} \\
\mathrm{HBr} &\triangleq (\nu\,\ell)\left(\mathrm{H}^{(\ell)} \mid \mathrm{Br}^{(\ell)}\right)
\end{aligned}$$

the affinity network,



,

the kinetic law,

$$\mathrm{L}_k\left([\mathrm{H}_2]\,,[\mathrm{Br}_2]\,,[\mathrm{HBr}_2]\right) = \frac{[\mathrm{H}_2]\,[\mathrm{Br}_2]^{1/2}}{1 + k\frac{[\mathrm{HBr}]}{[\mathrm{Br}_2]}},$$

and the process,

$$\Pi \triangleq [\mathrm{Br}_2] \cdot \mathrm{Br}_2 \parallel [\mathrm{H}_2] \cdot \mathrm{H}_2 \parallel [\mathrm{HBr}] \cdot \mathrm{HBr}.$$

When we apply the process semantics, we see we have the process vector,

$$\begin{aligned}
|\Pi\rangle &= [\mathrm{H}_2]\,|\mathrm{H}_2\rangle + [\mathrm{Br}_2]\,|\mathrm{Br}_2\rangle + [\mathrm{HBr}]\,|\mathrm{HBr}\rangle \\
&= [\mathrm{H}_2]\,|\mathrm{H}_2 \to (\ell, m)(\mathrm{H}^{(\ell)}|\mathrm{H}^{(m)})\rangle|h\rangle \\
&\quad + [\mathrm{Br}_2]\,|\mathrm{Br}_2 \to (\ell, m)(\mathrm{Br}^{(\ell)}|\mathrm{Br}^{(m)})\rangle|b\rangle \\
&\quad + [\mathrm{HBr}]\,|\mathrm{HBr} \to \mathrm{HBr}\rangle|h^*, b^*\rangle,
\end{aligned}$$

and we can compute the evolution vector as,

$$\begin{aligned}
\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t} &= \mathrm{L}_k(\mathrm{conc}_h\,|\Pi\rangle, \mathrm{conc}_b\,|\Pi\rangle, \mathrm{conc}_{h^*,b^*}\,|\Pi\rangle)\,\mathrm{React}\,(\mathrm{direct}_h\,|\Pi\rangle, \mathrm{direct}_b\,|\Pi\rangle, \mathrm{direct}_{h^*,b^*}\,|\Pi\rangle) \\
&= \mathrm{L}_k([\mathrm{H}_2]\,, [\mathrm{Br}_2]\,, [\mathrm{HBr}])\,\mathrm{React}\,\Big(|\mathrm{H}_2 \to (\ell, m)(\mathrm{H}^{(\ell)}|\mathrm{H}^{(m)})\rangle, \\
&\qquad |\mathrm{Br}_2 \to (\ell, m)(\mathrm{Br}^{(\ell)}|\mathrm{Br}^{(m)})\rangle, |\mathrm{HBr} \to \mathrm{HBr}\rangle\Big) \\
&= \frac{[\mathrm{H}_2]\,[\mathrm{Br}_2]^{1/2}}{1 + k\frac{[\mathrm{HBr}]}{Br2}}\,(2|\mathrm{HBr}\rangle - |\mathrm{H}_2\rangle - |\mathrm{Br}_2\rangle)
\end{aligned}$$

we then see that this is equivalent to the system of differential equations,

$$\frac{\mathrm{d}\,[\mathrm{H}_2]}{\mathrm{d}t} = \frac{\mathrm{d}\,[\mathrm{Br}_2]}{\mathrm{d}t} = -\frac{[\mathrm{H}_2]\,[\mathrm{Br}_2]^{1/2}}{1 + k\frac{[\mathrm{HBr}]}{Br2}} \qquad \frac{\mathrm{d}\,[\mathrm{HBr}]}{\mathrm{d}t} = 2\frac{[\mathrm{H}_2]\,[\mathrm{Br}_2]^{1/2}}{1 + k\frac{[\mathrm{HBr}]}{Br2}}.$$

Since the definition of the evolution vector depends on the total concentration of every instance of a each site involved in a reaction, it only make sense globally, and cannot easily be decomposed to give a compositional semantics. However, Kwiatkowski thesis showed it was possible to give a compositional semantics for Continuous-π, making heavy use of the fact that all reactions are given as unary or binary mass action reactions. The following result shows that our semantics can be reduce to similar compositional definitions in the special case that every reaction is a unary or binary mass action reaction.

Proposition 4.6.12 (Mass action special case)**.** *Suppose the affinity network $\mathcal{M}$ consists entirely of unary or binary mass action interactions. Then the definition of $\frac{\mathrm{d}}{\mathrm{d}t}$ has the following inductive definition as a special case,*

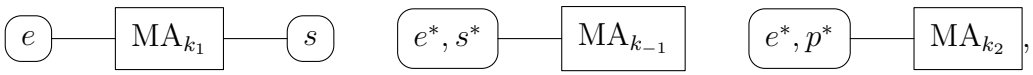
$$\begin{aligned}
\frac{\mathrm{d}(\gamma|S \to S'\rangle|\alpha\rangle)}{\mathrm{d}t} &= \Big(\textstyle\sum_{\mathrm{MA}_k(\alpha)\in\mathcal{M}} k\gamma\Big)\,\mathrm{React}\,|S \to S'\rangle \\
&\qquad + \frac{1}{2}\gamma^2|S \to S'\rangle|\alpha\rangle \oplus |S \to S'\rangle|\alpha\rangle \\
\frac{\mathrm{d}(|\Pi\rangle + |\Phi\rangle)}{\mathrm{d}t} &= \frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t} + \frac{\mathrm{d}|\Phi\rangle}{\mathrm{d}t} + |\Pi\rangle \oplus |\Phi\rangle
\end{aligned}$$

where $|\Pi\rangle, |\Phi\rangle$ *are orthogonal, and the* interaction tensor $\oplus$ *is defined on pure tensors by,*

$$|T_\alpha\rangle|\alpha\rangle \oplus |T_\beta\rangle|\beta\rangle = \sum_{\mathrm{MA}_k(\alpha,\beta)\in\mathcal{M} \text{ or } \mathrm{MA}_k(\beta,\alpha)\in\mathcal{M}} k\,\mathrm{React}(|T_\alpha\rangle, |T_\beta\rangle)$$

and extended bilinearly to all of $\mathbb{P}_\mathcal{M}$.

Example 4.6.13. We will check that Proposition 4.6.12 gives the same result we derived for the enzyme system in Example 4.6.9. Here we have



and,

$$|\Pi\rangle = |\Pi_1\rangle + |\Pi_2\rangle + |\Pi_3\rangle + |\Pi_4\rangle,$$

where,

$$|\Pi_1\rangle = [S]\,|S \to (\ell)S^*_\ell\rangle|s\rangle \qquad |\Pi_2\rangle = [E]\,|E \to (\ell)E^*_\ell\rangle|e\rangle$$
$$|\Pi_3\rangle = [C]\,|C \to S|E\rangle|s^*, e^*\rangle \qquad |\Pi_4\rangle = [C]\,|C \to P|E\rangle|p^*, e^*\rangle.$$

Now we may now apply the definition to calculate the combined evolution vector as,

$$\begin{aligned}
\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t} &= \frac{\mathrm{d}|\Pi_1\rangle}{\mathrm{d}t} + \frac{\mathrm{d}(|\Pi_2\rangle + |\Pi_3\rangle + |\Pi_3\rangle)}{\mathrm{d}t} + |\Pi_1\rangle \oplus (|\Pi_2\rangle + |\Pi_3\rangle + |\Pi_4\rangle) \\
&= \left(\frac{\mathrm{d}|\Pi_2\rangle}{\mathrm{d}t} + \frac{\mathrm{d}(|\Pi_3\rangle + |\Pi_4\rangle)}{\mathrm{d}t} + |\Pi_2\rangle \oplus (|\Pi_2\rangle + |\Pi_3\rangle + |\Pi_4\rangle)\right) \\
&\qquad + k_1\,[E]\,[S]\,(|C\rangle - |E\rangle - |S\rangle) \\
&= \left(\frac{\mathrm{d}|\Pi_3\rangle}{\mathrm{d}t} + \frac{\mathrm{d}|\Pi_4\rangle}{\mathrm{d}t} + |\Pi_3\rangle \oplus |\Pi_4\rangle\right) + k_1\,[E]\,[S]\,(|C\rangle - |E\rangle - |S\rangle) \\
&= k_{-1}\,[C]\,(|E\rangle + |S\rangle - |C\rangle) \\
&\qquad + k_2\,[C]\,(|E\rangle + |P\rangle - |C\rangle) \\
&\qquad + k_1\,[E]\,[S]\,(|C\rangle - |E\rangle - |S\rangle)
\end{aligned}$$

so we can see that the result agrees with our calculations in Example 4.6.9.

4.7 Implementation

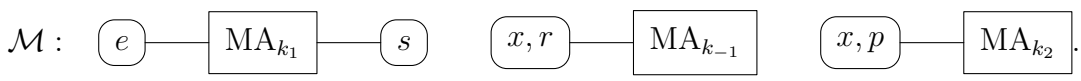
We have also developed an implementation of the calculus, using the Haskell programming language. This makes it possible to load models and perform simulations. We will now briefly describe the operation of the implementation. In order to simulate a model we

first need to translate it into a computer readable (plain text) syntax; the uses model files consisting of lists of definitions for each species, process, and kinetic law, along with a flat description of the affinity network as a list of names of kinetic laws along with the lists of bags of sites to which they are applied. The tool can then parse a model file, and convert the model to an abstract syntax tree, which is placed into normal form (along with De Bruijn indicies to assign bound locations unique names by α-conversion), to ensure that congruent agents are given a unique representation. We then use the process semantics to calculuate a vector representation of agents; we have implemented a custom vector library to allow vectors to be indexed by arbitrary datatype in the language and composed into tensor products, so we can represent the tensors of transitions between prime species and bags of sites upon which the semantics relies). This then makes it possible to calculate the evolution vector $\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t}$ for any mixture of species, and hence, to simulate the system using a standard numerical method for differential equations (in particular we use an adaptive Adams-Bashford/Adams-Moulten multistep predictor-corrector method).

In contrast to continuous π, we do not explicitly convert the model to a system of differential equations before simulation, rather, directly using the evolution vector to perform the simulation. Our approach is more straightforward to implement, and has the advantage of making it possible to simulate systems with infinite state spaces such as polymers (which correspond to infinite dimensional systems of differential equations), however, it does limit the performance of simulation as we are unable to use more efficient standard implementations of vectors and numerical methods. Therefore, further work is ongoing to add support for automatically extract systems of ODEs in the cases where this is possible.

Example 4.7.1 (Enzymes)**.** The syntax can be illustrated by recalling our enzyme example,

$$
\begin{aligned}
E &\triangleq e(l).x@l.E \\
S &\triangleq s(l).(r@l.S + p@l.P). \\
P &\triangleq \mathbf{0} \\
\Pi &\triangleq 0.1 \cdot E \parallel 1.0 \cdot S
\end{aligned}
$$



This translates into a computer readable model by first expressing the species definitions as,

```
species E = e(l) -> x@l -> E;

species S = s(l) -> p@l -> P;

species P = 0;
```

the affinity network as,

```
affinity network M(k1,m1,k2) = {
  e, s at rate MA(k1);
  x + r at rate MA(m1);
  x + p at rate MA(k2);
}
```

and, finally, the process as,

```
process Pi = [0.1] E || [1.0] S with network M(1.0, 0.1, 0.5);
```

(setting the rate constants to $k_1 = 1.0, k_{-1} = 0.1$, and $k_2 = 0.5$).

Then saving the complete model as `enzyme.biocpi`, we are able to simulate it with the biowb command line tool. The biowb tool gives a command prompt where we can enter various commands for loading and simulating models. This allows us to first load the model with

```
load models/enzyme.biocpi
```

and plot it from $t = 0$ to $t = 50$ with,

```
plotUptoEpsilon Pi 0 50 0.001 0.001 0.00001 0.0000001
```

(specifying an error tolerance of 0.001, an initial stepsize of 0.001, a minimum stepsize of 0.0001, and a minimum species population of 0.00000001). The simulation results can be seen in Figure 4.3 (top).

Example 4.7.2 (Michaelis-Menten Enzyme model)**.** We can also construct another model of our enzyme system using Michaelis-Menten kinetics, by specifying a custom kinetic law as part of the model.

This gives a model,

```
species S = s -> P;

species P = 0;

species E = e -> E;

kinetic law MichaelisMenten(k, K; C, S) = k*C*S/(K + S);

affinity network M(v,k) = {
  e, s at rate MichaelisMenten(v, k);
}

process Pi = [0.1] E || [1.0] S with network M(1.0, 2.0);
```

When we run the model we can see this gives similar results to the mass action model (see Figure 4.3).

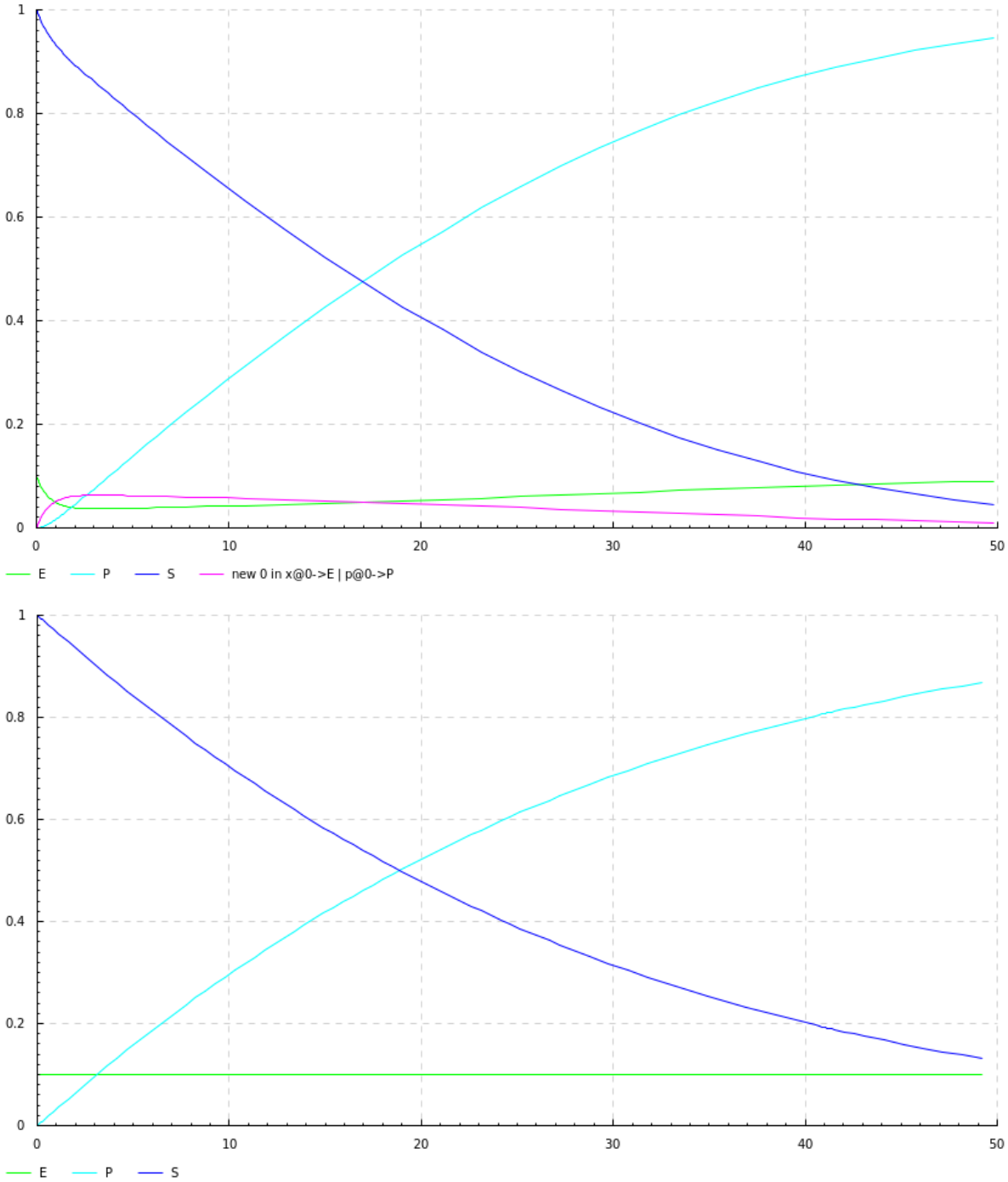


Figure 4.3: *Enzyme simulation results, using mass action kinetics (top) and Michaelis-Menten kinetics (bottom).*

Chapter 5

Applications in Biological and Chemical Modelling

In this chapter we demonstrate the ability of our language to model real biological and chemical systems. First we will present general expressiveness results for autonomous differential equations, and for chemical reactions networks. Then we will see some case studies modelling real biological systems: the Ping-Pong mechanism for enzymatic reactions, Lotka-Volterra Predator-Prey models, Kuznetsov's model of tumour immune interactions, and Elowitz and Stanislas' Repressilator.

5.1 Expressiveness results

The goal underlying all of the changes in this project was to build a biological process calculus general enough to express the full variety of interaction dynamics present in existing continuous state models of biological systems. In this section, we will show we have met this goal by presenting straightforward translations of the two most popular modelling frameworks, differential equations and chemical reaction networks, into our modelling language.

5.1.1 General encoding of autonomous dynamical systems

It can be seen quite easily that this extended language is sufficiently general to encode all autonomous dynamical equations. Suppose we have a system of differential equations,

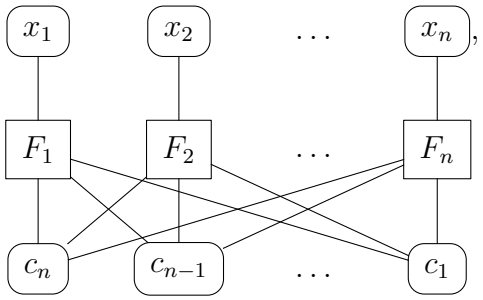
$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}(t)), \qquad \mathbf{x}(\mathbf{0}) = \mathbf{y}. \tag{a}$$

This can then be encoded as a continuous bond model with kinetic laws,

$$F_j([x_j], [c_1], \ldots, [c_n]) = f_j([c_1], \ldots, [c_n]),$$

with the affinity network,

$\mathcal{M}$:



(so that each F_j is connected up to x_j and $c_1, \ldots, x_n$ in that order), the species,

$$X_j \triangleq x_j.(X_j|X_j) + c_j.X_j \qquad\qquad (j \in \{1, \ldots, n\})$$

and process definition,

$$\Pi = y_1 \cdot X_1 \parallel y_2 \cdot X_2 \parallel \ldots \parallel y_n \cdot X_n.$$

It is worth noting that the rate laws used in this encoding may give negative values, so, for example, if the rate is negative the transitions $X_j \to X_j \mid X_j$ will destroy X_j rather than creating it. If this unnerves the reader, a more complicated translation could be give which avoid negative rates by splitting each component rate function f_j into the difference of two non-negative rate functions so $f_j = f_j^+ - f_j^-$ and adding separate actions to create and destroy each component.

We then have the following result,

Theorem 5.1.1 (Correctness of encoding)**.** *The system of differential equations associated with* Π *is equivalent to (a). That is, the vector field,*

$$\frac{\mathrm{d}|\Phi\rangle}{\mathrm{d}t} \qquad\qquad (\Phi = x_1 \cdot X_1 \parallel \ldots \parallel x_n \cdot X_n)$$

is isomorphic to the vector field defining (a).

Proof. For any Φ we find that,

$$\begin{aligned}\frac{\mathrm{d}|\Phi\rangle}{\mathrm{d}t} &= \sum_j F_j(\mathrm{conc}_{x_1}|\Phi\rangle, \mathrm{conc}_{c_1}|\Phi\rangle, \ldots \mathrm{conc}_{c_n}|\Phi\rangle) \\ &\qquad \mathrm{React}(\mathrm{direct}_{x_1}|\Phi\rangle, \mathrm{direct}_{c_1}|\Phi\rangle, \ldots \mathrm{direct}_{c_n}|\Phi\rangle) \\ &= \sum_j F_j(x_j, x_1, \ldots, x_n) \\ &\qquad \mathrm{React}(|X_j|X_j\rangle - |X_j\rangle, |X_1\rangle - |X_1\rangle, \ldots, |X_1\rangle - |X_1\rangle) \\ &= \sum_j f_j(x_1, \ldots, x_n)|X_j\rangle,\end{aligned}$$

which matches (a) if we identify $\mathbf{x} = (x_1, \ldots, x_n)$ with $|\Phi\rangle = x_1|X_1\rangle + \ldots + x_n|X_n\rangle$. □

5.1.2 Encoding chemical reactions networks

Given we have a general translation from differential equations into continuous-π, we know that in particular, every chemical reaction network can be represented. However, this encoding is very general, and does not capture the structure of the specific reaction network in question. Therefore, it will be instructive to try and give a more direct translation from chemical reaction networks into the continuous bond-calculus.

We will start by defining more formally what we mean by a chemical reaction network.

Definition 5.1.2. A *chemical reaction network* $\mathcal{N} = (\mathcal{S}, \mathcal{R}, \mathcal{K})$ consists of:

- A finite set $\mathcal{S} = \{S_1, \ldots, S_N\}$ of *species*.
- A finite set $\mathcal{R} = \{y_k \to y'_k\}_k \subseteq \mathbb{Z}^N_{\geq 0} \times \mathbb{Z}^{\mathcal{C}}_{\geq 0}$ of *reactions*, with *reaction vectors* $y'_k - y_k \in \mathbb{Z}^N$.
- A *kinetics* $\mathcal{K} : \mathcal{R} \to \mathbb{R}^N_{\geq 0} \to \mathbb{R}_{\geq 0}$, which associates to each reaction a *rate function*, which gives the reaction rate given the concentration of each species.

That is, a chemical reaction network involving species $\mathcal{S}$ and complexes of species C, consists of reactions $\mathcal{R}$, a relation $\mathcal{R}$ of reactions, each of which associates a complex of species representing the *reactants*, with another complex of species representing the *products*.

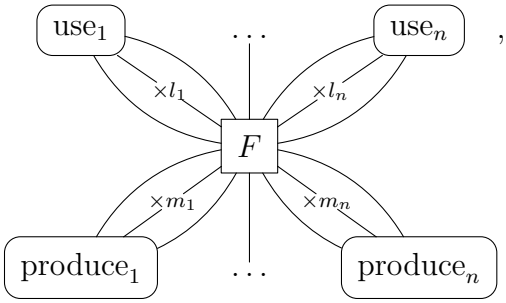
Our encoding $\mathcal{T}$ from chemical reaction networks to continuous bond-calculus models is as follows. Suppose we have a chemical reaction network $\mathcal{N} = (\mathcal{S} = \{A_1, \ldots, A_n\}, \mathcal{R}, \mathcal{K})$, then for each species A_j, we give $\mathcal{T}(\mathcal{N})$ a process,

$$\mathcal{T}(A_j) = \mathrm{use}_j.\mathbf{0} + \mathrm{produce}_j.(\mathcal{T}(A_j)|\mathcal{T}(A_j)).$$

Furthermore, for each reaction $\mathcal{R}$ of form,

$$\mathcal{R}:\ l_1 A_1 + l_2 A_2 + \ldots + l_n A_n \xrightarrow{\mathbf{f}} m_1 A_1 + m_2 A_2 + \ldots + m_n A_n$$

we add to the affinity network $\mathcal{M}_{\mathcal{N}}$ a component,



,

where we define the kinetic law $F = \mathcal{T}(f)$ as,

$$\begin{aligned} &F\Big(\,[\text{use}_1],\ldots,[\text{use}_1],\ldots,[\text{use}_n],\ldots,[\text{use}_n],\ldots,\\ &\qquad [\text{produce}_1],\ldots,[\text{produce}_1],\ldots,[\text{produce}_n],\ldots,[\text{produce}_n]\,\Big)\\ &= f\Big(\,[\text{use}_1],\ldots,[\text{use}_n]\,\Big). \end{aligned}$$

Finally, we define the process $\mathcal{T}(\mathcal{N})$ as,

$$\mathcal{T}(\mathcal{N}) \triangleq [A_1]\cdot\mathcal{T}(A_1) \parallel [A_2]\cdot\mathcal{T}(A_2) \parallel \ldots \parallel [A_n]\cdot\mathcal{T}(A_n).$$

We can see how this encoding works by considering a simple example,

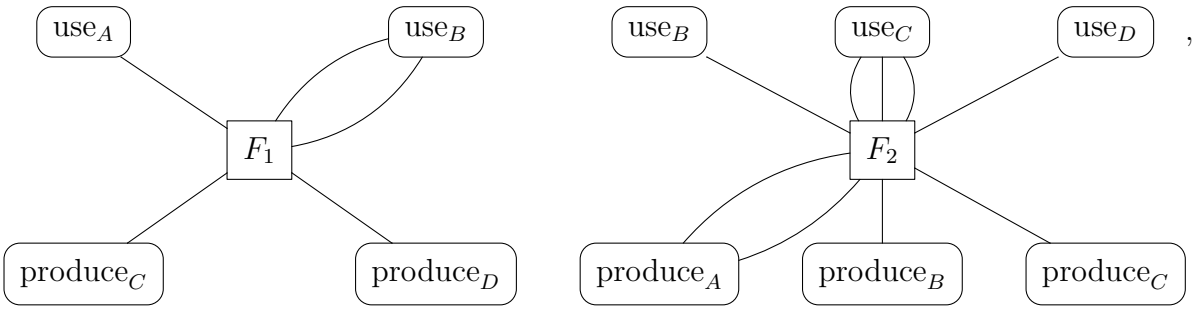
Example 5.1.3. Consider the chemical reaction network,

$$\begin{aligned} A \;+\; 2\,B \;&\xrightarrow{f_1}\; C \;+\; D\\ B \;+\; 3\,C \;+\; D \;&\xrightarrow{f_2}\; 2\,A \;+\; B \;+\; C\,. \end{aligned}$$

This is mapped to a continuous-π model with species,

$$\begin{aligned} \mathcal{T}(A) &\triangleq \text{use}_A.\mathbf{0} + \text{produce}.(\mathcal{T}(A) \mid \mathcal{T}(A))\\ \mathcal{T}(B) &\triangleq \text{use}_B.\mathbf{0} + \text{produce}.(\mathcal{T}(B) \mid \mathcal{T}(B))\\ \mathcal{T}(C) &\triangleq \text{use}_C.\mathbf{0} + \text{produce}.(\mathcal{T}(C) \mid \mathcal{T}(C))\\ \mathcal{T}(D) &\triangleq \text{use}_D.\mathbf{0} + \text{produce}.(\mathcal{T}(D) \mid \mathcal{T}(D)), \end{aligned}$$

affinity network $\mathcal{M}_{\mathcal{N}}$,



and process definition,

$$\Pi \triangleq [A] \cdot A \parallel [B] \cdot B \parallel [C] \cdot C \parallel [D] \cdot D.$$

We can now see that by looking at the system of differential equations associated with both the original chemical reaction network, and the continuous bond-calculus model associated with our transformation $\mathcal{T}$, that this model captures the dynamics of the original chemical reaction network.

Theorem 5.1.4 (Correctness of encoding)**.** *Let $\mathcal{D}$ be the map that takes chemical reaction networks to their underlying differential equations. Then for any chemical reaction network $\mathcal{N}$ we have that,*

$$\frac{\mathrm{d}\mathcal{T}(\mathcal{N})}{\mathrm{d}t} = \mathcal{D}(\mathcal{N}).$$

5.2 Modelling case studies

Given the general expressiveness results in the previous section, one might be tempted to stop here and declare our task done; after all, we already know that our language can express all of the systems we set out to model. However, just as Turing completeness is a rather weak recommendation for a programming language, we should not be content with a general expressiveness result before we try our hands at modelling some real biological systems in the language, and compare our results with existing models in the literature based on existing frameworks, with a critical eye for the effectiveness of each approach. We will now attempt to do just this, reviewing a wide range of biological and chemical models from the literature, and attempting to replicate the results in our framework. In looking at these models, we will first investigate how general kinetics allows us to make more concise and understandable models, and then compare our approach with other process algebras for biological modelling.

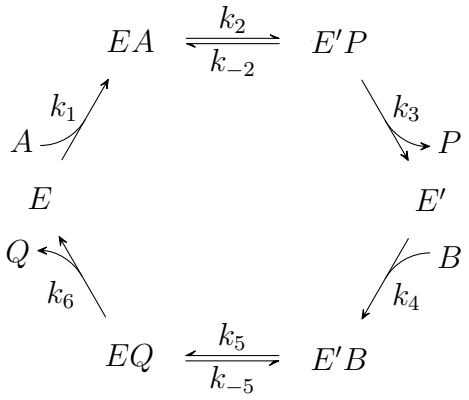


Figure 5.1: *The Ping-Pong Mechanism*

5.2.1 The Ping-Pong mechanism

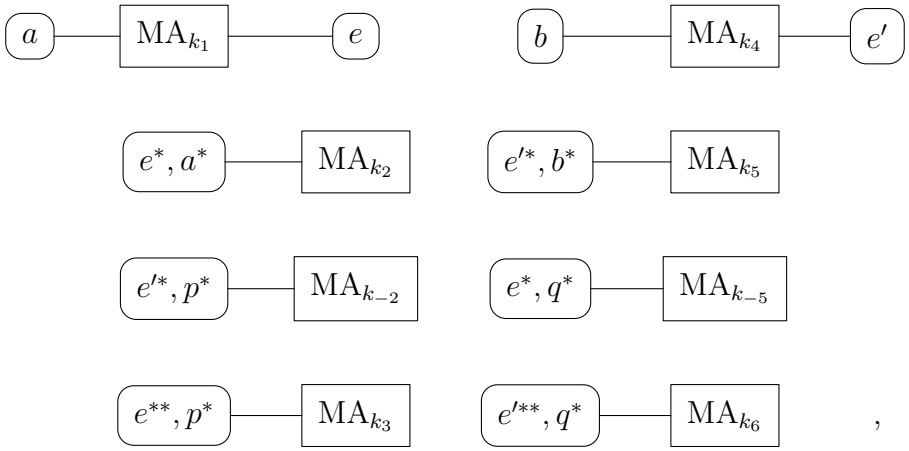
When developing the language we looked at some examples of enzymatic reactions, using Michaelis-Menton Kinetics. Whist these had relatively simple mechanisms, many enzymes have much more complicated mechanisms, and we will investigate one of these, the Ping-Pong mechanism [82], as a realistic example of using process algebra to understand reaction mechanisms. The Ping-Pong reaction starts similarly to a Michaelis-Menten reaction, in that we have a substrate A and an enzyme E which bind together to form a complex (although now the binding is irreversible), allowing it to be transformed into a product P and then released. The key difference is that here our enzyme, rather than coming out of the reaction unchanged, changes into a different state E' which then acts an enzyme for a second stage of the reaction where substrate B binds and is transformed to release product Q, and, importantly, return the enzyme back to state E, allowing the reaction cycle to continue.

Mass action model

We can model this system using the following species definitions,

$$
\begin{aligned}
E &\triangleq e(\ell).E^*_\ell & E' &\triangleq e'(\ell).E'^*_\ell \\
E^*_\ell &\triangleq e^*@\ell.E'^*_\ell + e'^{**}@\ell.E & E'^* &\triangleq e'^*@\ell.E^*_\ell + e'^{**}@\ell.E' \\
A &\triangleq a(\ell).A_\ell & B &\triangleq b(\ell).B_\ell \\
A^*_\ell &\triangleq a^*@\ell.P_\ell + a^{**}@\ell.A & B^*_\ell &\triangleq b^*@\ell.Q_\ell + b^{**}@\ell.B \\
P^*_\ell &\triangleq p^*@\ell.A^*_\ell + p^{**}@\ell.P & Q^*_\ell &\triangleq q^*@\ell.B^*_\ell + q^{**}@\ell.Q \\
P &\triangleq p.P & Q &\triangleq q.Q
\end{aligned}
$$

the affinity network,



,

and process definition,

$$\Pi \triangleq [A] \cdot A \parallel [E] \cdot E \parallel [B] \cdot B \parallel [E'] \cdot E'.$$

The enzymes and substrates will also form the dynamic complexes,

$$EA \triangleq (\nu\, \ell)(E^*_\ell \mid A^*_\ell), \qquad E'B \triangleq (\nu\, \ell)(E'^*_\ell \mid B^*_\ell),$$
$$EQ \triangleq (\nu\, \ell)(E^*_\ell \mid Q^*_\ell), \qquad E'P \triangleq (\nu\, \ell)(E'^*_\ell \mid P^*_\ell).$$

The result of simulating this model is shown in Figure 5.2 (top).

Michaelis-Menten model

It is also possible to apply a Michaelis-Menten style approximation, modelling the whole reaction via the kinetic law,

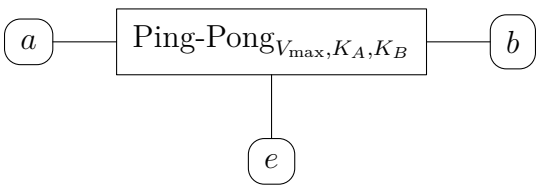
$$\text{Ping-Pong}_{V_{\max},K_A,K_B}([A]\,,[B]\,,[E]) \triangleq \frac{V_{\max}\,[A]\,[B]\,[E]}{K_A\,[B] + K_B\,[A] + [A]\,[B]},$$

where $V_{\max}$ is the maximum reaction velocity, K_A is the equilibrium constant for the reaction $EA \rightleftharpoons E'P$, and K_B is the equilibrium constant for the reaction $E'B \rightleftharpoons EQ$.

Using this approximation we may build a simplified model with agents,

$$A \triangleq a.P \qquad B \triangleq b.Q \qquad E \triangleq e.E$$
$$P \triangleq p.P \qquad Q \triangleq q.Q,$$

the affinity network,



and process definition,

$$\Pi \triangleq [A] \cdot A \parallel [B] \cdot B \parallel [E] \cdot E.$$

This model represents the whole ping-pong reaction as a multiway interaction between the substrates A and B and the enzyme E, governed by the Ping-Pong kinetic laws. A simulation result from this model can be seen in Figure 5.2 (bottom).

Model evaluation

In the mass action version of the model we see that sites locations provide an effective way to represent the dynamic formations underlying the Ping-Pong mechanism. In the version of the model using Michaelis-Menten kinetics, we how the combination of multiway synchronisation and general kinetics make it possible to significantly simplify a model, and whilst reducing the number of parameters from 8 to 3. In Figure 5.2 we see a fairly good agreement between the two models, however, the Michaelis-Menten model fails to capture the lag between the two reactions,

$$A \xrightarrow{\text{E}\ \text{E'}} P \qquad \text{and} \qquad B \xrightarrow{\text{E'}\ \text{E}} Q,$$

caused by the delay between the start of the first reaction, and the availability of the modified enzyme E' for the second reaction. There is another limitation of the approximate model that we only see when considering its behaviour as part of a larger system. In making the approximation we have removed a number of sites and the intermediate complexes, hiding the mechanism driving the reaction; this means that the approximation will not necessarily hold when the model is embedded into a larger biological system with extra interactions involving these hidden sites and complexes.

We should now ask how how easily other languages could be applied to modelling this mechanism. We are not aware of any other process algebra models of the Ping-Pong mechanism, however, other process algebras including Bio-PEPA, Continuous-π, and Stochastic-π have considered models of simpler enzymatic reactions. The mass action version of the model could be represented in continuous-π, however, the model would be more complicated, requiring one global and two local affinity networks (one for each enzyme). The mass action version of the model could be represented similarly in Stochastic-π, which would allow stochastic simulation. However, both Stochastic-π and Continuous-π could not represent the simplified version of the model due to the presence of multiway interactions and general kinetics. The simplified version of the model would

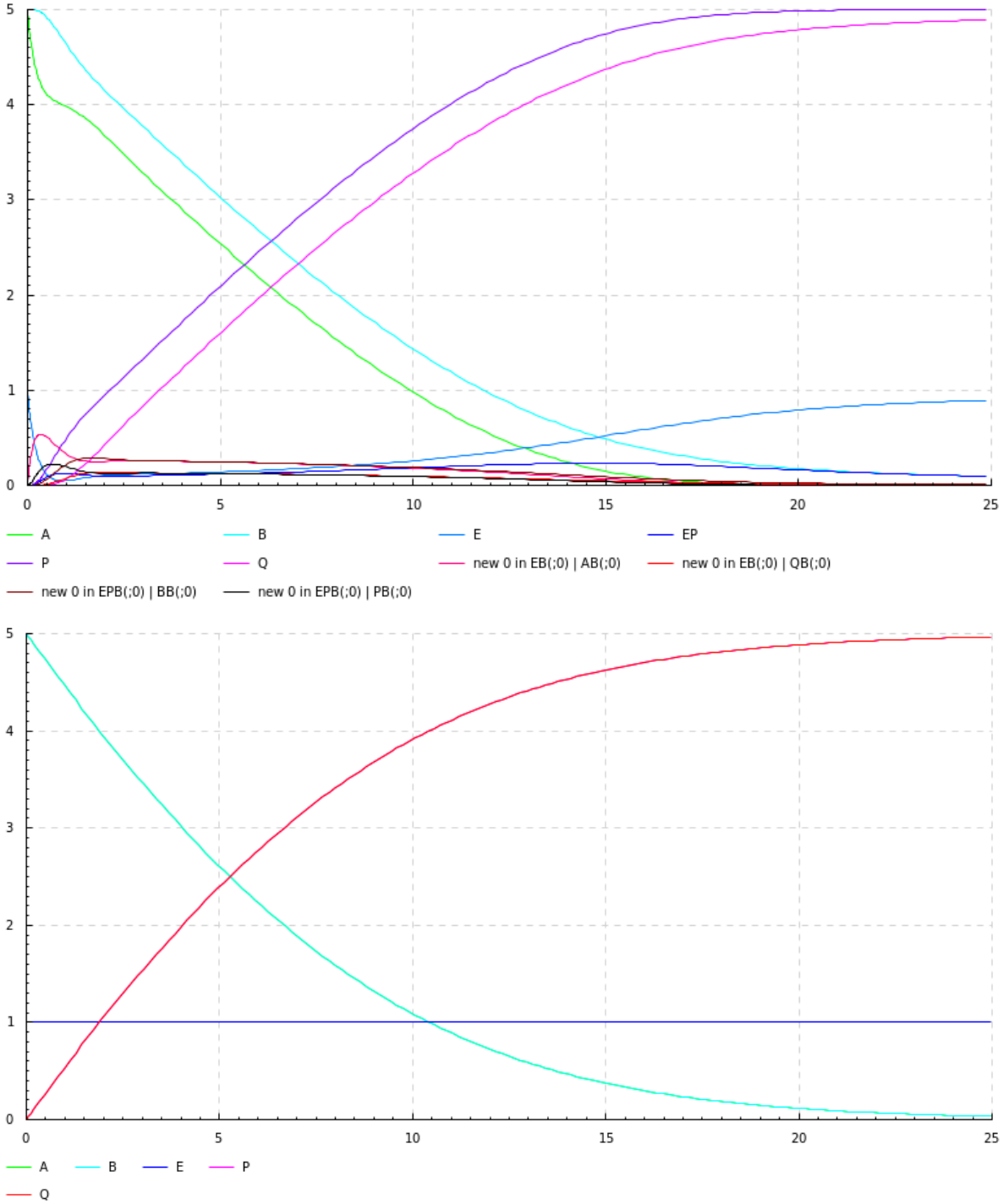


Figure 5.2: *Ping-Pong mechanism simulation results, with mass action kinetics (top), and Michaelis-Menten kinetics (bottom). The mass action simulation has parameters* $k_1 = k_{-1} = k_4 = k_{-5} = 1, k_2 = k_5 = 2, k_3 = k_6 = 3$*, whilst the Michaelis-Menten simulation has the corresponding parameters* $V_{\max} = 1, K_A = K_B = 2$*.*

translate directly to a Bio-PEPA model, and this would allow both stochastic simulation and ODE extraction. However, in order to represent the mass action version of the model, Bio-PEPA would need for extra species for each of intermediate complexes to be specified explicitly. Custom languages have also been developed for modelling enzyme mechanisms, including kMech [119] which is implemented as a Maple plugin, and supports analysing mathematical models of pathways. As a special purpose language for analysing enzyme mechanisms, kMech has constructs to concisely model a variety of mechanisms including the Ping-Pong mechanism, and to derive differential equations. Whilst such tools may be more practical in specific cases, we still believe it is worth investigating enzyme mechanisms within a more general framework such as process algebra.

5.2.2 Lotka-Volterra Predator-Prey models

In this section we will be interested in modelling population dynamics in an ecosystem of interacting species in a predator prey relationship. We can model such a scenario by representing each species of our ecosystem with a species in the model, and modelling interspecies interactions as communication. The use of dynamical systems for population modelling is well established – many of the modelling assumption and rate laws we use standard in the area, and are described more fully in texts and undergraduate courses including [46].

Base model

Suppose for example, we have a species R of rabbits, whom are being preyed upon by a species of very hungry foxes F. The growth of the rabbits is explained for the ability of each rabbit to reproduce, producing two rabbits[1]. Foxes can reproduce in a similar manner, but need to eat a rabbit to do so, causing the death of the rabbit in the process. Finally, foxes can die of natural causes. This description is enough to define the species of the system,

$$\text{Rabbit} \triangleq \text{reproduce}.(\text{Rabbit} \mid \text{Rabbit}) + \text{beEaten}.\mathbf{0} \qquad \text{Fox} \triangleq \text{eat}.(\text{Fox} \mid \text{Fox}) + \text{die}.\mathbf{0}$$

and the global affinity network,

[1]We model growth by asexual reproduction since this simplification is standard in population modelling.

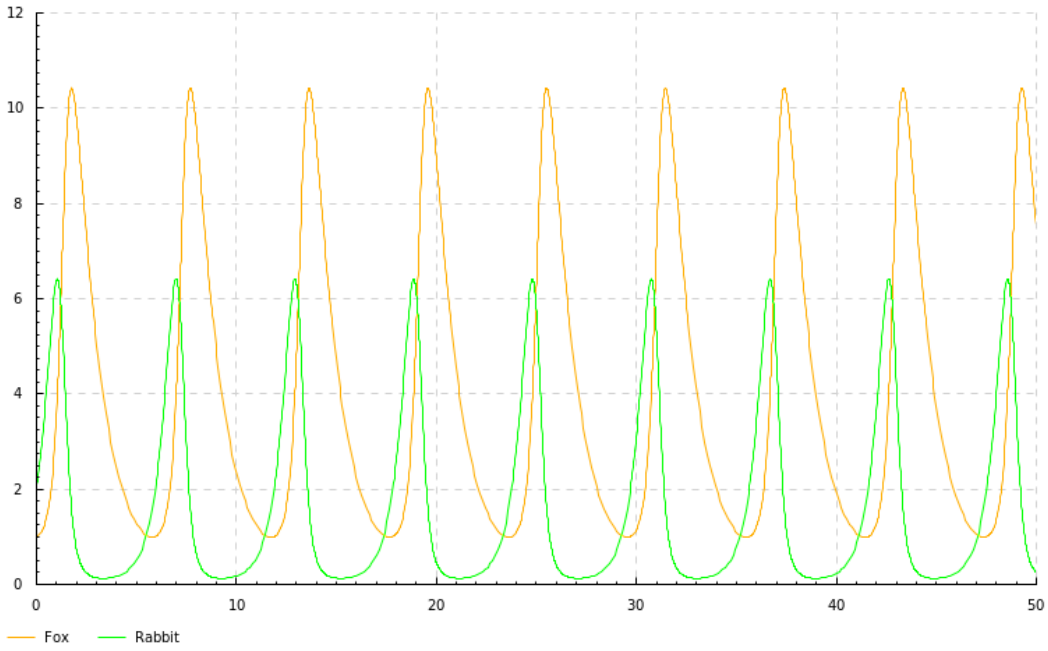


Figure 5.3: *Mass action population simulation with* $\alpha = 2.0, \beta = 0.5, \gamma = 0.8$.

where rates of reproduction, consumption, and fox death are given by mass action kinetics with stoichiometric rates α, β, γ respectively. This use of the law of mass action corresponds to the assumption that the rate at which each fox/rabbit reproduces or dies is independent of the rest of the population. If we convert this system to ODEs, we get the following equations,

$$\begin{aligned} \frac{\mathrm{d}\,[\text{Rabbit}]}{\mathrm{d}t} &= \alpha\,[\text{Rabbit}] - \beta\,[\text{Fox}]\,[\text{Rabbit}] \\ \frac{\mathrm{d}\,[\text{Fox}]}{\mathrm{d}t} &= \beta\,[\text{Fox}]\,[\text{Rabbit}] - \gamma\,[\text{Fox}]\,. \end{aligned}$$

which one may recognise as the classic Lotka-Volterra equations for interspecies competition. An example simulation is in Figure 5.3, which shows oscillation between rabbit population and fox population. This demonstrates we have a formal way of deriving many classical population models from agent behaviour, making all of the assumptions underlying the model clear.

Functional rates

Of course, this model is rather unrealistic as we assume that the species can carry out all of these actions at a rate independent of the current population; we are assuming that rabbits can carry on reproducing forever, and that the effectiveness of hunting does

not depend on the rabbit population. It is then very natural to extend the model with more sophisticated kinetic laws which more accurately model the rates of growth and predation. For example, in order to model the dependency of the effectiveness of hunting of the fox and rabbit populations, we can replace the law for the rate of predation with,

$$\text{Functional}_{\beta,h}(F,R) = \frac{\beta F R}{1+\beta h R}$$

where we introduce a new parameter, h, the *handling time*. We can also better model the growth of the rabbits with the logistic law,

$$\text{Logistic}_{\alpha,k}(R) = \alpha R(1+R/k)$$

which suggests that the population initially grows exponentially at rate b, but slows as it approaches its carrying capacity k, which represents the largest population the environment can sustain. Uses these laws on a revised affinity network now produce the systems of differential equations,

$$\begin{aligned}\frac{\mathrm{d}\left[\text{Rabbit}\right]}{\mathrm{d}t} &= \alpha\left[\text{Rabbit}\right]\left(1+\frac{1}{k}\left[\text{Rabbit}\right]\right) - \frac{\beta\left[\text{Fox}\right]\left[\text{Rabbit}\right]}{1+h\left[\text{Rabbit}\right]}\\ \frac{\mathrm{d}\left[\text{Fox}\right]}{\mathrm{d}t} &= \frac{\beta\left[\text{Fox}\right]\left[\text{Rabbit}\right]}{1+h\left[\text{Rabbit}\right]} - \gamma\left[\text{Fox}\right].\end{aligned}$$

Simulation results are shown in Figure 5.4, which display similar qualitative results, but a flatter shape of response curve. Logistic rabbit growth has less of an effect than in an isolated rabbit growth model, since predation already places a limit on the rabbit population. Note that we are able to improve the accuracy of our model by changing affinity network to incorporate kinetic laws which better model the data, without changing the structure of the model.

Multicoloured rabbits

Now what if we had two prey species? Suppose our rabbits now come in two varieties, red rabbits R, and blue rabbits B. They are distinct species, so we have to keep track of their populations individually. However, a hungry fox does not care what colour of rabbit it eats; be it red, blue, or aqua marine, any rabbit is just as tasty. This time we will use a functional rate for fox predation, and logistic growth for rabbits. One tricky aspect of the model this that whilst the two populations of rabbits cannot breed, they both eat the same food so the carrying capacity applies to the total population of rabbits across both species, leading us to the following modified version of the logistic rule,

$$\text{Logistic}_{\alpha,k}(\left[\text{reproduce}\right],\left[\text{consumeResources}\right]) \triangleq \alpha\left[\text{reproduce}\right]\left(1-\frac{1}{k}\left[\text{consumeResources}\right]\right)$$

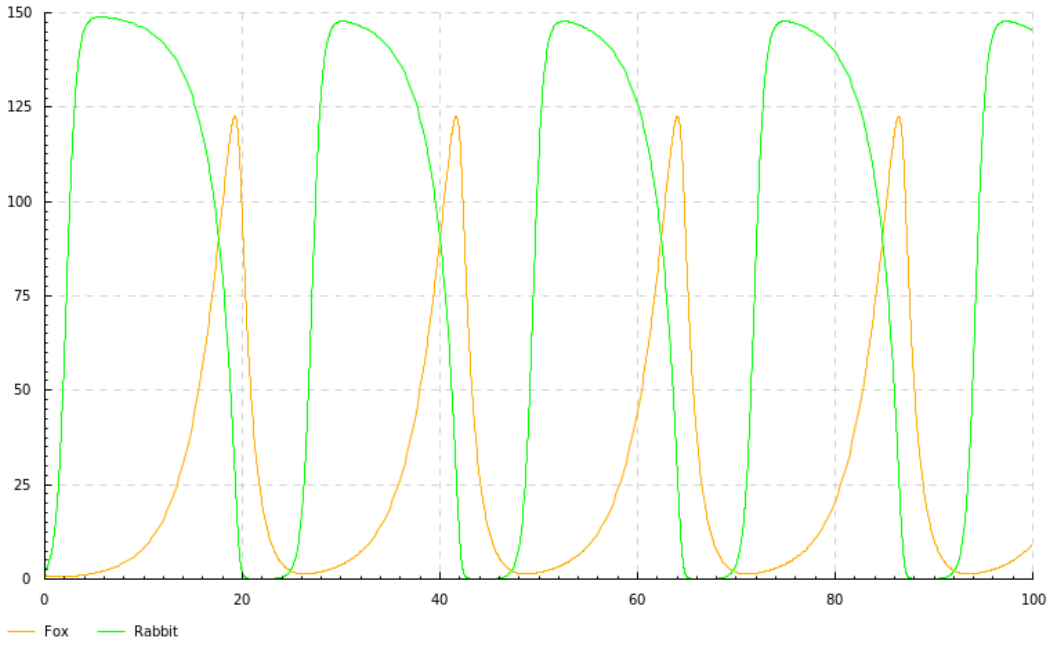


Figure 5.4: *Simulation for logistic and functional rates with* $\alpha = 2.0, k = 150, \beta = 0.08, h = 0.8, \gamma = 0.8$.

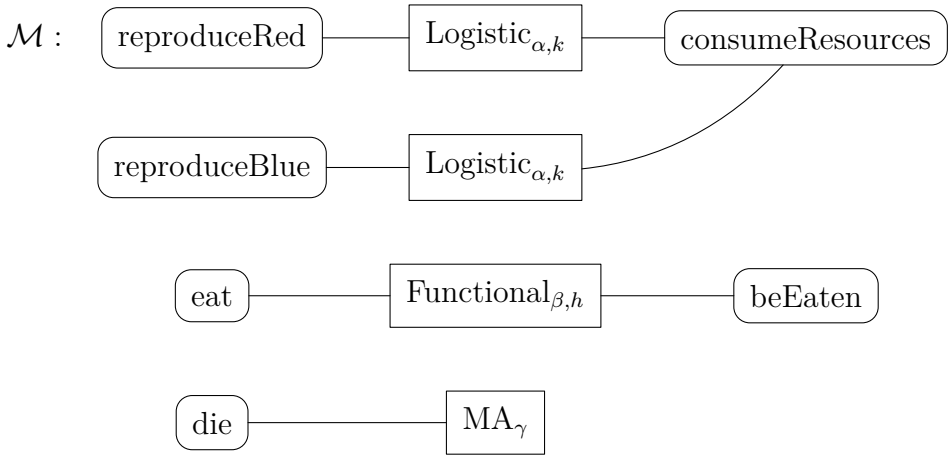
which says that for a given species, the growth rate is proportional to the concentration of reproducing agents [reproduce], whilst this carrying capacity is based on the concentration of resource consumers [consumeResources] across as species which are in direct competition (i.e. that eat the same kind of food).

This suggests to us the following model,

$$\text{Red} \triangleq \text{reproduceRed}.(\text{Red} \mid \text{Red}) + \text{consumeResources}.\text{Red} + \text{beEaten}.\mathbf{0}$$

$$\text{Blue} \triangleq \text{reproduceBlue}.(\text{Blue} \mid \text{Blue}) + \text{consumeResources}.\text{Blue} + \text{beEaten}.\mathbf{0}$$

$$\text{Fox} \triangleq \text{eat}.(\text{Fox} \mid \text{Fox}) + \text{die}.\mathbf{0}$$



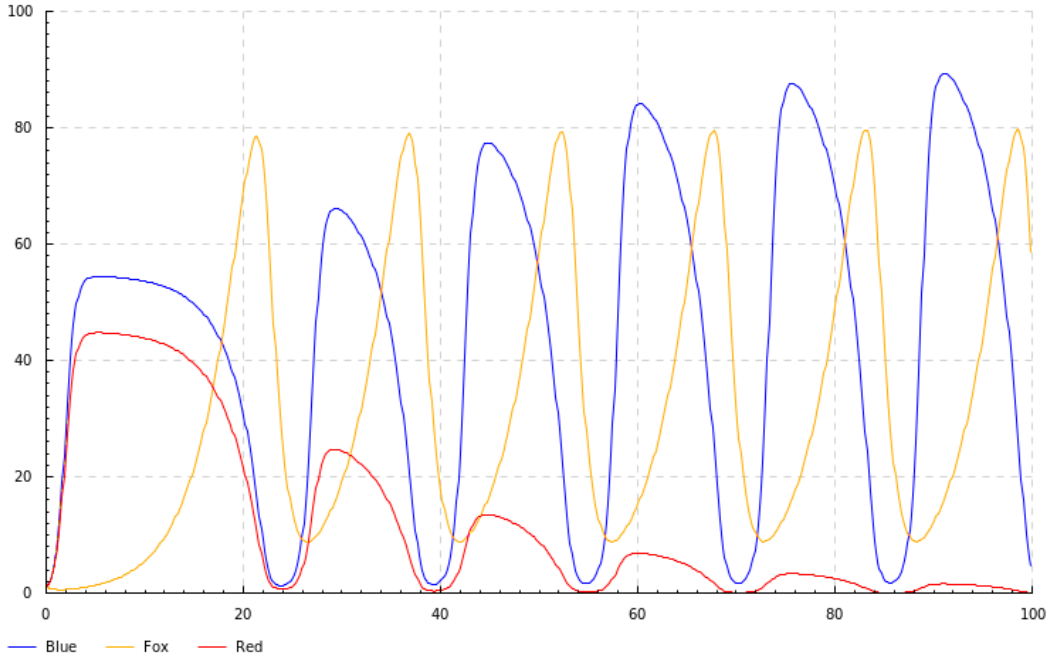


Figure 5.5: *Two population simulation with* $\alpha_{Red} = 2.0, k_{\text{Red}} = 100, l_{\text{Red}} = 10, \alpha_{Blue} = 2.1, k_{\text{Blue}} = 100, l_{\text{Red}} = 10, \beta = 0.08, h = 0.8, \gamma = 0.8.$

This model gives the system of differential equations,

$$\begin{aligned}
\frac{\mathrm{d}\,[\text{Red}]}{\mathrm{d}t} &= \alpha\,[\text{Red}]\left(1 - \frac{1}{k}([\text{Red}] + [\text{Blue}])\right) - \frac{\beta\,[\text{Fox}]\,[\text{Red}]}{1 + \beta h([\text{Red}] + [\text{Blue}])} \\
\frac{\mathrm{d}\,[\text{Blue}]}{\mathrm{d}t} &= \alpha\,[\text{Blue}]\left(1 - \frac{1}{k}([\text{Red}] + [\text{Blue}])\right) - \frac{\beta\,[\text{Fox}]\,[\text{Blue}]}{1 + \beta h([\text{Red}] + [\text{Blue}])} \\
\frac{\mathrm{d}\,[\text{Fox}]}{\mathrm{d}t} &= \frac{\beta\,[\text{Fox}]\,([\text{Red}] + [\text{Blue}])}{1 + \beta h([\text{Red}] + [\text{Blue}])} - \gamma\,[\text{Fox}]
\end{aligned}$$

Simulation results are shown in Figure 5.5, that one the competition with foxes results in only the fittest (fastest reproducing) species of rabbit surviving; the success of one population of rabbits supports a larger population of foxes which will also target their less fortunate neighbours. This model demonstrates the importance of the relationship between site and species, and the care we took in calculating the concentration and direction of a process at a site, in capturing that simply splitting our rabbit population into two different colours does not change the way the growth rate for foxes depends on the total rabbit population.

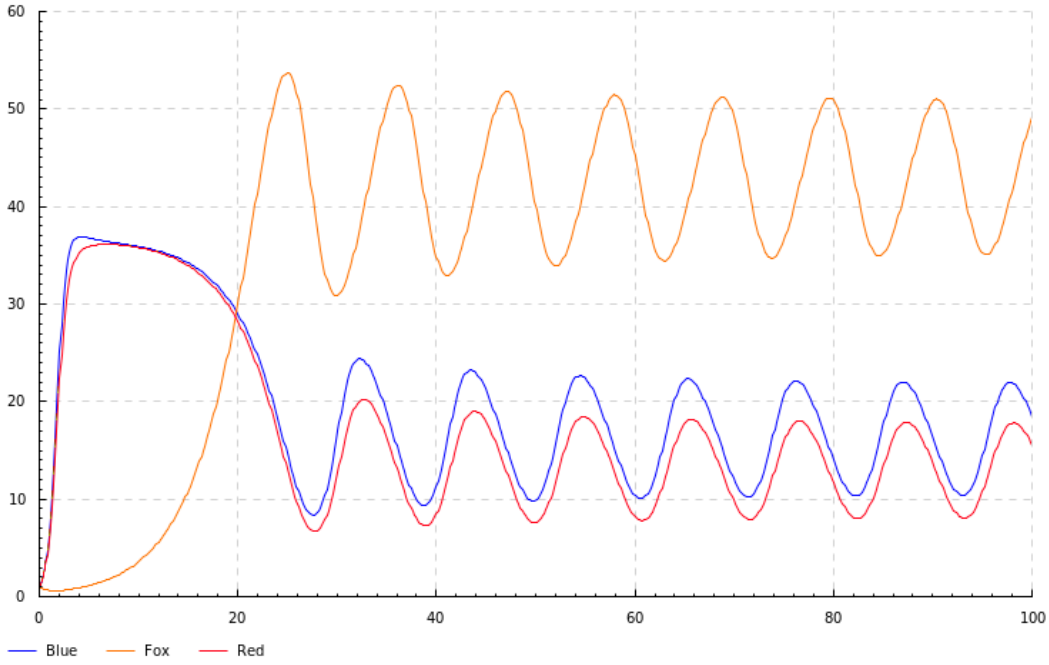


Figure 5.6: *Rabbits and grass simulation with* $\alpha_{Red} = 2.0, k_{\text{Red}} = 100, \alpha_{Blue} = 2.1, k_{\text{Blue}} = 100, \beta = 0.08, h = 0.8, \gamma = 0.8$.

Coloured rabbits meet coloured grass

For one final variation on this model, we suppose that along side the general food source, there is a small amount of red and blue grass, which only the corresponding species of rabbit can eat. This corresponds to the revised logistic law,

$$\text{Logistic}_{\alpha,k,l}([\text{reproduce}], [\text{consumeResources}]) \triangleq$$
$$\alpha\,[\text{reproduce}]\left(1 - \frac{[\text{consumeResources}] + [\text{reproduce}]}{k + l}\right)$$

Using this law on a revised affinity network produces Figure 5.6, in which both species survive, and their population oscillates with the fox population – this provides a good demonstration of the importance of an evolutionary niche for the long term survival of a species.

Model evaluation

The models in this section show the applicability of our language to developing population models. We note the effectiveness of affinity networks and functional rules in this case, as they allow us to refine a model in stages to integrate more sophisticated modelling assumption. The relationship between sites and agents allows us to reflect that a species

may play many different roles, or that the same role may be played by multiple species, and proves effective in modelling competition of species for shared resources or predators targeting multiple prey species. The continuous-π calculus has not previously been applied to population modelling, and its restriction to mass action kinetics would not allow any but our most basic model to be described. Therefore, these examples show that the advances in our semantics bring a new range of applicability in population modelling. In developing our series of models, we have also seen that biologically interesting changes in the relationship between species, often result in a small change to a single component of the model, but results a larger change to the resulting differential equations, suggesting that the process algebra model better captures the structure of the biological system.

Other process algebras have previously been applied to population modelling. In [1], a Stochastic-π model of the Lotka-Volterra system was considered, producing similar oscillatory solutions to our model, however the restriction to mass action kinetics would prevent any but the basic model being expressed. This considered both continuous and stochastic simulation, and, as showed the stochastic model shows similar dynamics with more variability. Since Bio-PEPA is designed for biological modelling it was originally conceived as a conservative process algebra respecting conservation of mass, making it difficult to represent birth/death processes, however, Bio-PEPA for Epidemiological Models [34] removed this restrictions, and introduced a number of features suitable for population modelling. This means that Bio-PEPA would be able to effectively represent all of the models we have considered in this section. WSCCS (Weighted Stochastic CCS) has also been applied to population modelling, including deriving differential equations from agent behaviour [90, 84]. Bio-PEPA [34] and other languages such as MELA [117] are also able to model other features of populations such as spatial structure which we have not considered in these models.

5.2.3 Kuznetsov's model of immunogenic tumour growth

Mathematical modelling is a significant part of efforts to better understand how tumours develop, and to improve treatments. Mathematical models can reveal general patterns in tumour response to different treatment regimes, and are increasingly being developed to predict patient specific treatment responses; one survey is [54].

Whilst many tumours evolve mechanisms to evade immune response to various degrees, immune interactions still play a significant part in understanding tumour growth and treatments, with the effectiveness of treatments often depending on the dynamics of

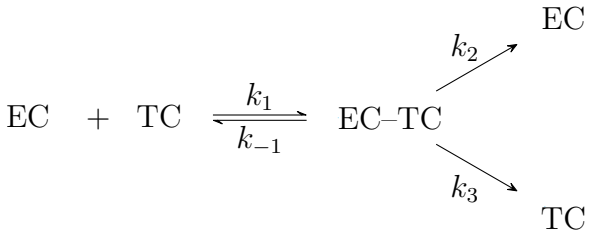


Figure 5.7: *Reaction scheme for Tumour Cell/Immune Cell interactions,* in vitro.

these interactions. Furthermore, immunotheropy seeks to devise new treatments which enhance the immune response, and has recently shown promising results in a range of clinical settings [56, 113, 108]. Therefore there has been a lot of interest in modelling the dynamics of tumour immune interactions. One of the most influential models of this type is Kuznetsov's model of immunogenic tumour growth [79] which models tumour cells and immune effector cells as two species, with effector cells binding to tumour cells resulting in one of three outcomes as depicted in Figure 5.7: either they unbind unchanged, they unbind killing the tumour cell, or they unbind killing the effector cell. The immune response is described by the functional law,

$$s + \frac{f\,[\mathrm{C}]}{g + [\mathrm{TC}]}$$

which includes a constant response rate s, which represents the base level of immune activity, as well as an adaptive response which is proportional to the number of complexes C, but limited by the number of unbound tumour cells. Tumour growth is described by the logistic law,

$$a\,[\mathrm{TC}]\,(1 - b([\mathrm{TC}] + [\mathrm{C}]))$$

which is proportional to the concentration of tumour cells, but limited by the carrying capacity $1/b$ which applies to the concentration of both bound and unbound tumour cells.

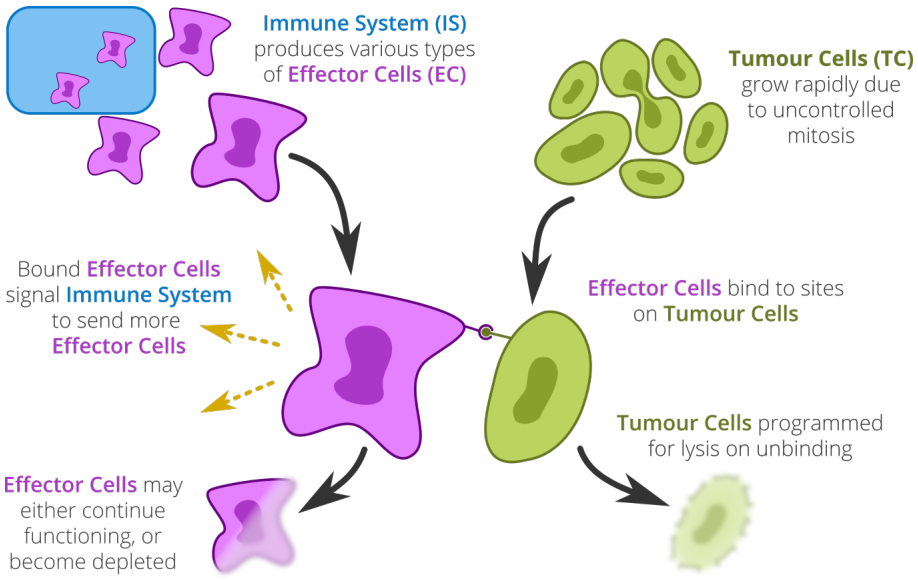


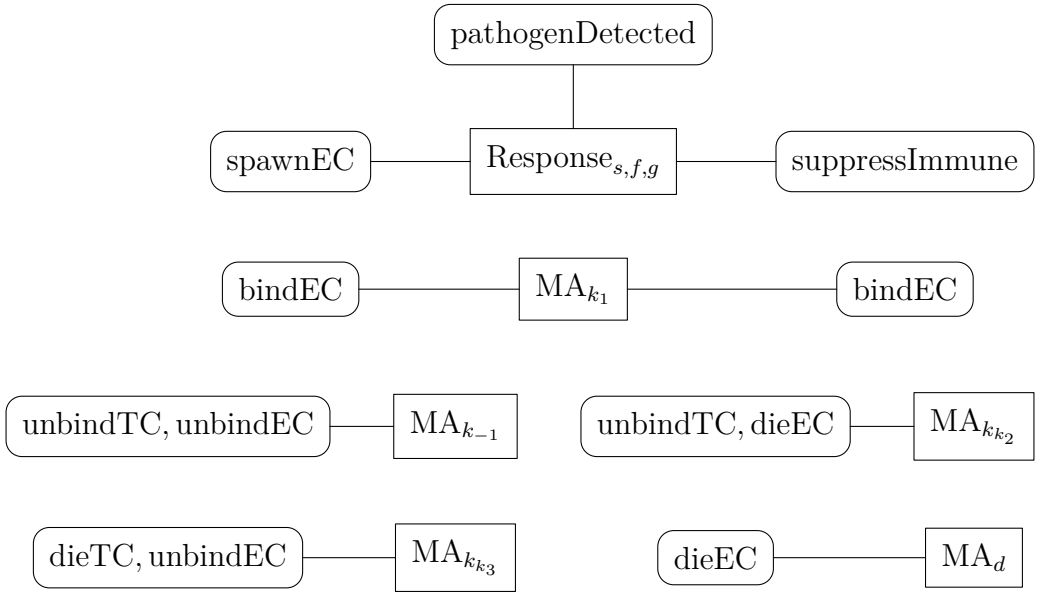
Figure 5.8: *The immune interactions underlying the model.*

Model with dynamic complexes

We are able to capture the Kuznetsov's model as agents performing the interactions shown in Figure 5.8, resulting in the following agents,

$$
\begin{aligned}
\mathrm{TC} &\triangleq \mathrm{growTC}.(\mathrm{TC} \mid \mathrm{TC}) + \mathrm{bindTC}(\ell).\mathrm{TC}^*_\ell \\
&\quad + \mathrm{suppressImmune}.\mathrm{TC} + \mathrm{consumeResources}.\mathrm{TC} \\
\mathrm{TC}^*_\ell &\triangleq \mathrm{unbindTC@}\ell.\mathrm{TC} + \mathrm{dieTC@}\ell.\mathbf{0} \\
&\quad + \mathrm{consumeResources}.\mathrm{TC}^*_\ell \\
\mathrm{EC} &\triangleq \mathrm{bindEC}(\ell).\mathrm{EC}^*_\ell + \mathrm{dieEC}.\mathbf{0} \\
\mathrm{EC}^*_\ell &\triangleq \mathrm{unbindEC@}\ell.\mathrm{EC} + \mathrm{dieEC@}\ell.\mathbf{0} \\
&\quad + \mathrm{pathogenDetected}.\mathrm{EC}^*_\ell \\
\mathrm{IS} &\triangleq \mathrm{spawnEC}.(\mathrm{IS} \mid \mathrm{EC})
\end{aligned}
$$

the affinity network,



the kinetic law definitions,

$$\text{Logistic}_{a,b}([\text{growTC}]\,,[\text{consumeResources}]) \triangleq a\,[\text{growTC}]\,(1-[\text{consumeResources}])$$

$$\text{Response}_{s,f,g}([\text{actEC}]\,,[\text{actTC}]\,,[\text{spawnEC}]) \triangleq s + \frac{p\,[\text{pathogenDetected}]}{g+[\text{suppressImmune}]}$$

and finally, the process,

$$\Pi \triangleq [\text{TC}] \cdot \text{TC} \parallel [\text{EC}] \cdot \text{EC} \parallel 1 \cdot \text{IS}.$$

This model consists of three types of agents, Tumour Cells TC, Effector Cells EC, and the wider Immune System IS. Binding is described using mass action kinetics, as is each of the fates of the effector/tumour cell complex. The immune response is modelled as a 3-way interaction between the immune system which synchronised on spawnEC to spawn new effector cells, the complexes which send the pathogenDetected signal, and the unbound tumour cells which send the suppressImmune signal.

From this models we can derive the following system of differential equations,

$$\begin{aligned}
\frac{\mathrm{d}\,[\text{EC}]}{\mathrm{d}t} &= s + \frac{f\,[\text{C}]}{g+[\text{TC}]} - d_1\,[\text{EC}] - k_1\,[\text{EC}]\,[\text{TC}] + (k_{-1}+k_2)\,[\text{C}] \\
\frac{\mathrm{d}\,[\text{TC}]}{\mathrm{d}t} &= a\,[\text{TC}]\,(1-b([\text{TC}]+[\text{C}])) - k_1\,[\text{EC}]\,[\text{TC}] + (k_{-1}+k_3)\,[\text{C}] \\
\frac{\mathrm{d}\,[\text{C}]}{\mathrm{d}t} &= k_1\,[\text{EC}]\,[\text{TC}] - (k_{-1}+k_2+k_3)\,[\text{C}]
\end{aligned}$$

which matches the mass action version of Kuznetsov's model. The results of simulating the model for one particular set of parameter values is shown in Figure 5.9. This shows oscillatory behaviour, demonstrating the possibility for tumour regrowth in the case the immune response is too slow or insufficiently sustained.

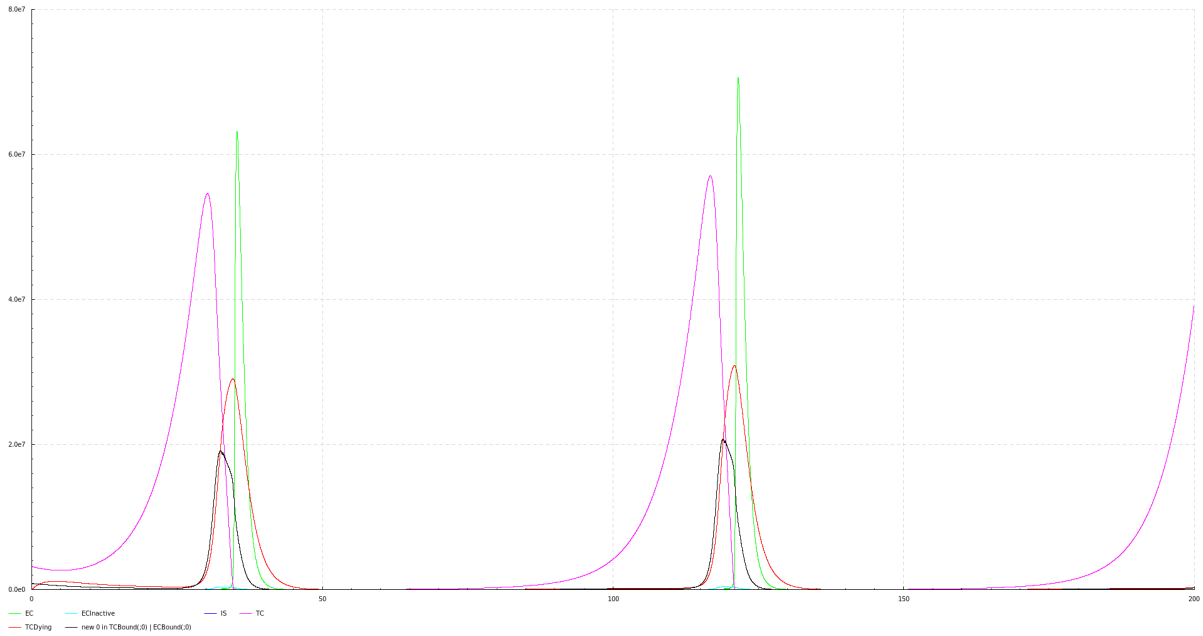


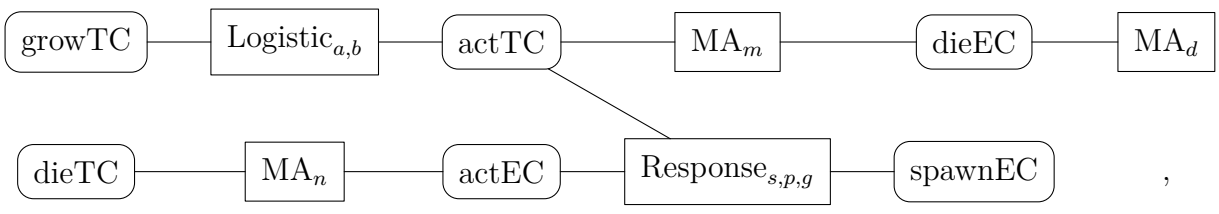
Figure 5.9: *A sample run of Kuznetsov's model with dynamic complexes, showing the possibility of sustained oscillations.*

Michaelis-Menten Model

Kuznetsov also considered a simplified version of the model using Michaelis-Menten kinetics to approximate the binding/unbinding of the effector cells to the tumour cells. He argued that this is a suitable approximation since in real tumour immune interactions there are usually many fewer complexes than tumour or effector cells [79], allowing the Michaelis-Menten approximation to apply in much the same way as it does in enzymatic reactions. A simplified version of our model using Michaelis-Menten kinetics is given by the agents,

$$\begin{aligned} \mathrm{TC} &\triangleq \mathrm{growTC}.(\mathrm{TC} \mid \mathrm{TC}) + \mathrm{dieTC}.\mathbf{0} + \mathrm{actTC}.\mathrm{TC} \\ \mathrm{EC} &\triangleq \mathrm{dieEC}.\mathbf{0} + \mathrm{actEC}.\mathrm{EC} \\ \mathrm{IS} &\triangleq \mathrm{spawnEC}.(\mathrm{IS} \mid \mathrm{EC}), \end{aligned}$$

the affinity network,



,

the kinetic laws,

$$\text{Logistic}_{a,b}([\text{growTC}]\,,[\text{actTC}]) \triangleq a\,[\text{growTC}]\,(1-[\text{actTC}])$$
$$\text{Response}_{s,f,g}([\text{actEC}]\,,[\text{actTC}]\,,[\text{spawnEC}]) \triangleq s + \frac{p\,[\text{actEC}]\,[\text{actTC}]}{g+[\text{actTC}]}$$

and the process,

$$\Pi \triangleq [\text{TC}] \cdot \text{TC} \parallel [\text{EC}] \cdot \text{EC} \parallel 1 \cdot \text{IS}.$$

This model reproduces the following model from the original paper,

$$\frac{\mathrm{d}\,[\text{EC}]}{\mathrm{d}t} = s + \frac{p\,[\text{EC}]\,[\text{TC}]}{g+[\text{TC}]} - m\,[\text{EC}]\,[\text{TC}] - d\,[\text{EC}]$$
$$\frac{\mathrm{d}\,[\text{TC}]}{\mathrm{d}t} = a\,[\text{TC}]\,(1-b\,[\text{TC}]) - n\,[\text{EC}]\,[\text{TC}]$$

An important feature of this reduced model is that it now has fewer parameters, and none of these parameters depend on the concentration of complexes. This made it possible for Kuznetsov to give values to these parameter based on biologically plausible values from the literature, and experiments in chimeric mice [79], as follows,

$$a = 0.18\ \text{day}^{-1}, \qquad b = 2.0 \times 10^{-9}\ \text{cells}^{-1},$$
$$s = 1.3 \times 10^{4}\ \text{cells day}^{-1} \qquad p = 0.1245\ \text{day}^{-1},$$
$$g = 2.019 \times 10^{7}\ \text{cells}, \qquad m = 3.422 \times 10^{-10}\ \text{day}^{-1}\ \text{cells}^{-1},$$
$$n = 1.101, \qquad d = 0.0412 \times 10^{-7}\ \text{day}^{-1}.$$

The results of simulating the model are shown in Figure 5.10. Here again we see oscillations, but this time damped, leading to a stable equilibrium, with the tumour surviving at a fixed size[2].

Model evaluation

These models demonstrate our ability to capture complex biological systems with multi-way interactions, general kinetics, and dynamic complex formation. We can also see the versatility of the abstract concepts underlying the language, in a new biological context, moving from molecular binding to binding of cells. The Michaelis-Menten has parameters

[2]We should note that this result cannot be directly compared with the results of the mass action model, since the parameters of that model have not been fitted to biological data. In the future it might, however, be interesting to fit the mass action model to the Michaelis-Menten model by the method described in Stanley Wang's dissertation [104] to assess the accuracy of Kuznetsov's approximation.

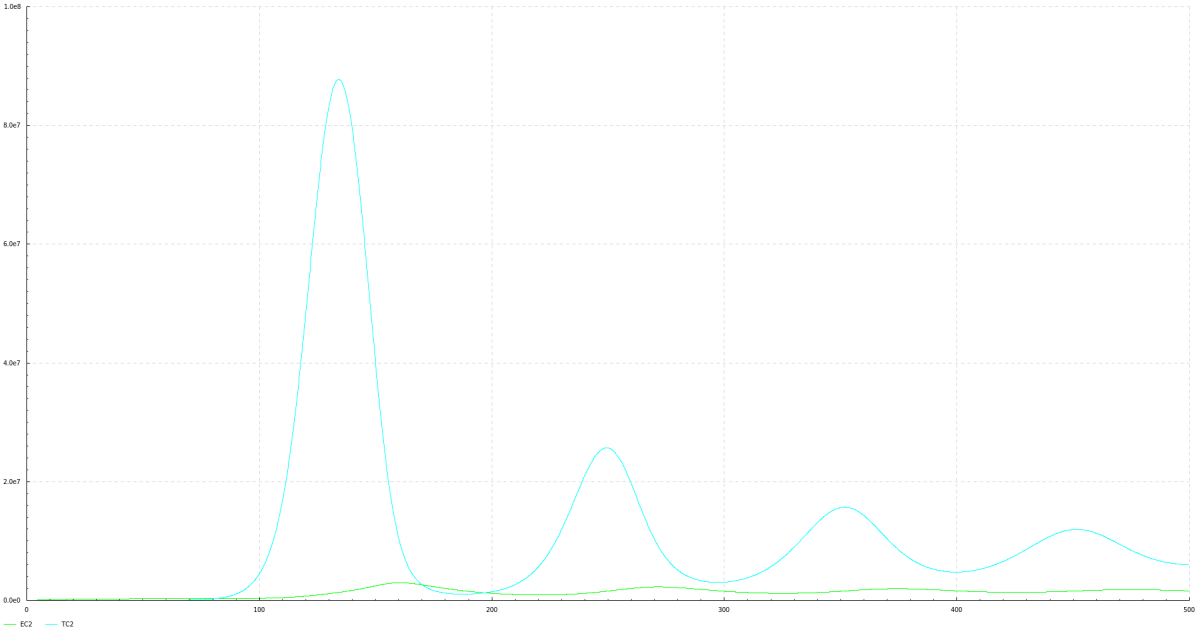


Figure 5.10: *A sample run of Kuznetsov's model using the Michaelis-Menten approximation, with parameters based on experimental data.*

based on experiments results in mice [79], showing us an example of the predictive use of mathematical models in real biological system, and the advantages of general kinetics in fitting models to experimental data.

We are not aware of any existing process algebra version of Kuznetsov's model, however, Bio-PEPAd has been applied to other models of tumour immune interactions [22, 23]. The Bio-PEPAd models also considered time lags in agent interactions (which translate to Delay Differential Equations, or stochastic simulation as a generalised semi-Markov process), which we cannot capture in our differential equation based semantics. It would also be possible to capture Kuznetsov's model in Bio-PEPA (with the non-conservative extension described in [34]), however, the complex would have to be represented explicitly. The heavy use of general kinetics and multiway synchronisation would make this model difficult to replicate in continuous-π or stochastic-π. In [58], a stochastic differential equation version of Kuznetsov's model was compared with agent based simulations, to assess the constituency of agent based simulation with the more established ODE models. Our model complements this work, by showing it is possible to derive the differential equations models, from a formal agent based description of the system.

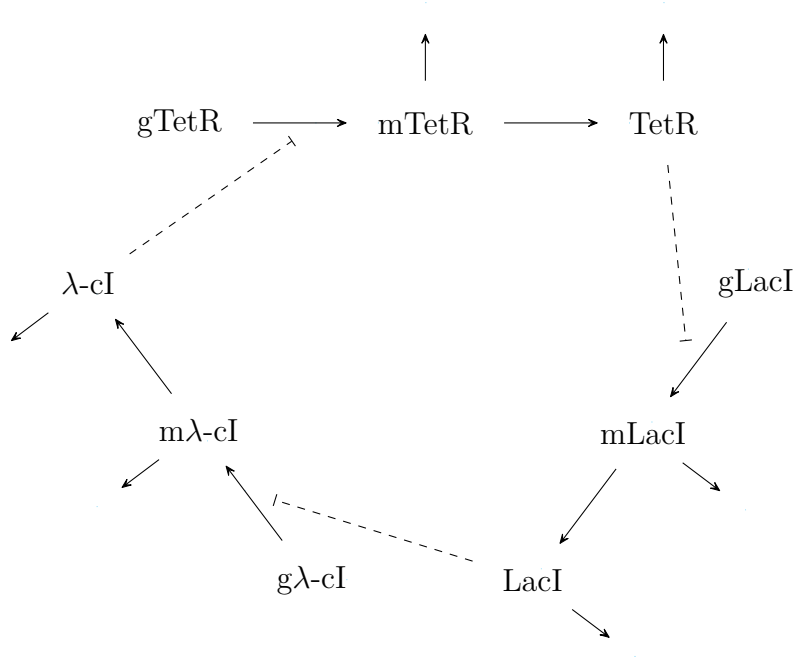


Figure 5.11: *Genetic circuit for Repressilator.*

5.2.4 Repressilator

Whilst most of the models we have discussed so far have been interested in describing living systems, we will now finish with an example from synthetic biology, a rapidly growing field which is interested not only in observing, but creating new biological entities, and has already produced novel biological implementations of logic gates, memory cells, cameras, and oscillators, using a combination of biochemistry and gene editing. With the advent of synthetic biology, came new demands for creating and analysing mathematical models in order to prototype and validate potential designs, without requiring costly experiments at every stage of development. The system we have chosen to investigate is the Repressilator [53], an artificial oscillator which Elowitz and Stanislas created as a genetic circuit, and then implanted as a DNA plasmid into the genome of the bacteria E. Coli using the λ-phage virus. This oscillator has became of the most recognisable case studies for synthetic biology, and there has been much interest in studying variants of the feedback mechanism, whilst it has been modelled in a variety of different modelling frameworks [33].

The Repressilator uses mRNA translations as its mechanism. We have three separate proteins – TetR, LacI, and λ-cI – which are first encoded by various sequences of DNA (their genes, denoted by gTetR, gLacI, and gλ-cI) , which are then transcribes into

$$\frac{\mathrm{d}\,[\mathrm{mTetR}]}{\mathrm{d}t} = -b\,[\mathrm{mTetR}] + \frac{a}{1+[\lambda\text{-cI}]} \qquad \frac{\mathrm{d}\,[\mathrm{TetR}]}{\mathrm{d}t} = b\,[\mathrm{mTetR}] - d\,[\mathrm{TetR}]$$
$$\frac{\mathrm{d}\,[\mathrm{mLacI}]}{\mathrm{d}t} = -b\,[\mathrm{mLacI}] + \frac{a}{1+[\mathrm{TetR}]} \qquad \frac{\mathrm{d}\,[\mathrm{LacI}]}{\mathrm{d}t} = b\,[\mathrm{mLacI}] - d\,[\mathrm{LacI}]$$
$$\frac{\mathrm{d}\,[m\lambda\text{-cI}]}{\mathrm{d}t} = -b\,[m\lambda\text{-cI}] + \frac{a}{1+[\mathrm{LacI}]} \qquad \frac{\mathrm{d}\,[\lambda\text{-cI}]}{\mathrm{d}t} = b\,[m\lambda\text{-cI}] - d\,[\lambda\text{-cI}]$$

Figure 5.12: *The ODEs for Repressilator model [53], where a is the transcription rate, d is the decay rate, and b is the translation rate. Note that the equations appear in a slightly different form in the paper, as there the variables are scaled to reduce the dimensionality of the model.*

sequences of mRNA (mTetR, mLacI, and mλ-cI), and finally, transcribed into the associated proteins. The oscillation comes from the fact that each of the proteins is in fact a repressor inhibiting the transcription of the next and leading to a negative feedback oscillator. The proteins and mRNA also decay from the system at a exponential rate[3]. The complete scheme of reactions for the oscillator is shown in Figure 5.11. The use of transcriptional regulation causes the system to exhibit nonlinear dynamics which are not describable using the law of mass action, since the rate of gene transcription, regulated by an inhibitor I is governed by the Hill equation,

$$R = \frac{k}{K + [I]^n}.$$

In the designing the Repressilator, Elowitz and Stanislas employed a continuous state differential equation model to explore the space of potential parameters, to choose a design which would lead to stable oscillators.

Our model

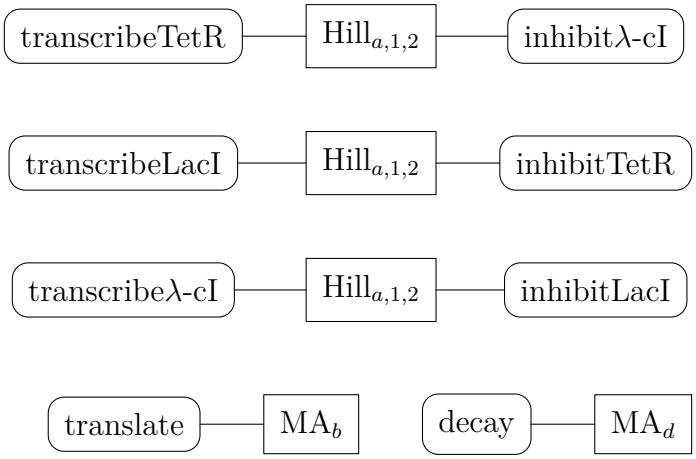
We can model the Repressilator with the following agents,

$$\mathrm{mTetR} \triangleq \mathrm{decay}.\mathbf{0} + \mathrm{translate}.\mathrm{TetR} \qquad \mathrm{TetR} \triangleq \mathrm{decay}.\mathbf{0} + \mathrm{inhibitTetR}.\mathrm{TetR}$$
$$\mathrm{mLacI} \triangleq \mathrm{decay}.\mathbf{0} + \mathrm{translate}.\mathrm{LacI} \qquad \mathrm{LacI} \triangleq \mathrm{decay}.\mathbf{0} + \mathrm{inhibitLacI}.\mathrm{LacI}$$
$$\mathrm{m}\lambda\text{-cI} \triangleq \mathrm{decay}.\mathbf{0} + \mathrm{translate}.\lambda\text{-cI} \qquad \lambda\text{-cI} \triangleq \mathrm{decay}.\mathbf{0} + \mathrm{inhibit}\lambda\text{-cI}.\lambda\text{-cI}$$

[3]Whilst easy to forget, the degradation of unused proteins and mRNA, is one of the most important processes in a cell, and can sometime have as large an effect on the dynamics of genetic circuits as synthesis.

$$\begin{aligned}\text{Genes} \triangleq\ & \text{transcribeLacI}.(\text{Genes} \mid \text{mLacI}) \\ & + \text{transcribeTetR}.(\text{Genes} \mid \text{mTetR}) \\ & + \text{transcribe}\lambda\text{-cI}.(\text{Genes} \mid \text{m}\lambda\text{-cI})\end{aligned}$$

the affinity network,



the kinetic law definition,

$$\text{Hill}_{k,K,n}(x, y) \triangleq \frac{k}{K + y^n}$$

and the process,

$$\begin{aligned}\Pi \triangleq\ & [\text{TetR}] \cdot \text{TetR} \parallel [\text{mTetR}] \cdot \text{mTetR} \parallel [\text{LacI}] \cdot \text{LacI} \parallel [\text{mLacI}] \cdot \text{mLacI} \\ & \parallel [\lambda\text{-cI}] \cdot \lambda\text{-cI} \parallel [\text{m}\lambda\text{-cI}] \cdot \text{m}\lambda\text{-cI} \parallel 1 \cdot \text{Genes}.\end{aligned}$$

In this model each process can decay at at stoichiometric rate d and mRNA is translated to protein at stoichiometric rate b. Gene transcription is modelled as communication between the proteins which send inhibition signals, and the Genes process which carries out the actual transcription; the kinetics of this reaction is governed by Hill kinetics and depends only on the inhibitor concentration.

Model evaluation

We used the implementation of our language to construct the model and perform simulations for a number of parameter values. Figure 5.13 shows a sample of the results – as expected the model shows a bifurcation between damped oscillations and sustained oscillations. We have confirmed that the simulation results match an ODE model of the dynamics, and that the semantics yield the correct system of differential equations, with the exception that we introduce an additional variable for the gene pool, and that the

form is slightly different. This extra variable is constant so does not impact simulation results, but it would also be possible to a model omitting it, at the expense of less explicitly modelling the components of the system. The Repressilator has been modelled in various other process algebras including Bio-PEPA [33], sCCP [15], and Stochastic-π [11, 1].

The Bio-PEPA model uses separate species to represent mRNAs and Proteins similarly to our model, and general kinetic laws. However, the specification of kinetic laws is somewhat different as the Hill kinetic laws had to be repeated for each species, whereas we use a single definition of the kinetic law, but reuse it for multiple species via the affinity network. Affinity networks also, also give our model a slightly different perspective, since the involvement of sites in reactions is specified globally in the network, rather than case by case in the agent definitions. The Bio-PEPA model was simulated using fluid flow approximation, showing similar oscillating behaviour to our model, and also via Gillespie's Stochastic Simulation Algorithm. The stochastic-π and sCCP models use a simpler model of the Repressilator, which defines a general negative regulation gene gate, and constructs a defines the Repressilator based on the reaction network. This gives a more compositional approach to modelling gene regulation networks, however, the models are based on mass action kinetics. Nevertheless, these models show oscillatory behaviour comparable to our model.

We can also ask to what extent the continuous models of the system match the stochastic models. Whilst in most cases the continuous models of the system oscillate in a similar manner to stochastic models, the stochastic model shows much more variability, with no fixed amplitude or period. Controlling reducing these stochastic fluctuations is an major design goal for synthetic oscillators, with significant amounts of work being carried out on designing more reliable oscillators [63, 97]. Continuous models are inherently unable to analyse such features of models suggesting the important of stochastic simulation in these cases. In [15], cases were considered where the stochastic system displays sustained oscillations, where the continuous model displays damped oscillations which soon die out. This further highlights the need for the user to aware of the limitations of continuous modelling approaches in feedback systems such as Repressilator which may be sensitive to small stochastic fluctuations.

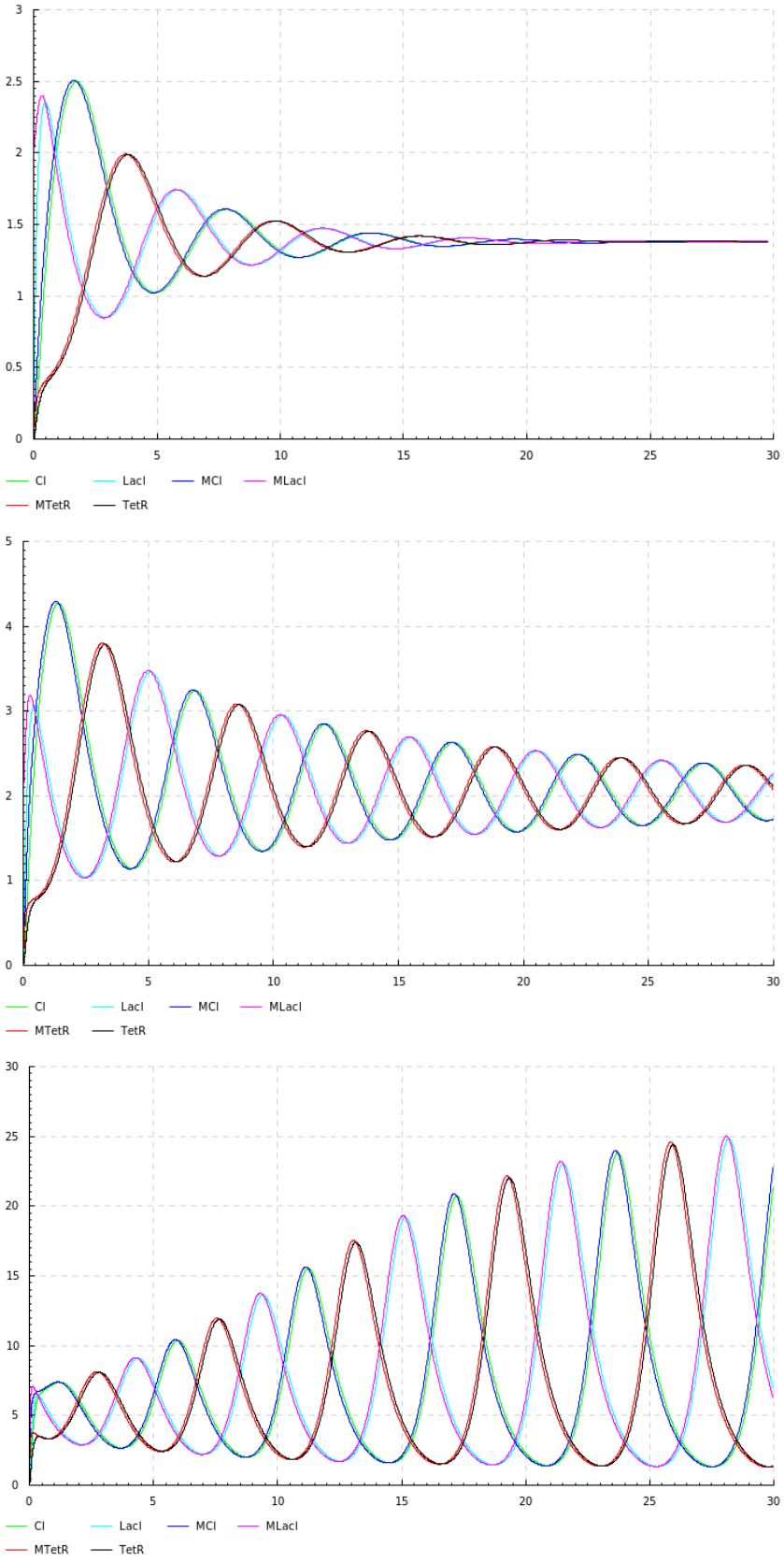


Figure 5.13: *Repressilator with $b = 10$, $d = 1$, and $a = 4, 14, 100$ from top to bottom.*

Chapter 6

Conclusion

At the start of this dissertation we set out to make two main extensions to the continuous π-calculus, to model multiway synchronisation and general reaction kinetics. In the intervening chapters we have presented a new construct for multiway synchronisation, investigated a number of qualitative calculi for multiway interactions, before settling of a new base calculus combining sites, locations, and compatibility networks. We then moved on to quantitative modelling, introducing rate laws, an extended version of affinity networks, and a two level calculus for fluid mixtures of species. From this we built a new transition semantics for potential interactions of species, and a process semantics for quantitative reactions of mixtures. Finally, we assessed the applicability of our new language, by presenting general encoding of autonomous dynamical systems and chemical reaction networks, along with a number of case studies modelling real biological systems.

6.1 Critical evaluation and related work

We must now ask ourselves to what extent our new calculi have met our original goals of modelling multiway synchronisation and general kinetics, and how it relates to existing process algebras which have considered these features.

Multiway synchronisation

In Sections 5.2, both Kuznetsov's tumour model and the Ping-Pong model make use of multiway synchronisations, demonstrate the ability of the language to capture multiway interactions in real biological systems. One potential criticism is that in these models the multiway synchronisation does not result in the creation of dynamic complexes, and

hence does not exercise our mechanism for coordination on shared locations. This is partially because in many models multiway communications are introduced as a way to hide the dynamic complexes in the underlying model, and so, existing models of isolated components of biochemical pathways have the tendency to give either a completely explicit mass action description of a mechanism, or condense it into a single rate law. However, we argue that the combination of multiway interactions and dynamic complex formation will be important in creating modular models of more complex reaction pathways, since these involve multiple stages each described by general kinetics, with the products not indivisible output species, but forming the inputs of subsequent reactions. This is an instance of a more general problem – building quantitative models large enough to see the benefits of process algebra's central features of modularity and compositionality, remains challenging due to the difficulty in fitting model parameters to real biological systems, and the difficulty of isolating components of biological systems, but the proliferation of such models be necessary to justify the overhead of process algebra over concise but non-composable modelling frameworks such as differential equations. We are currently working to prepare a model of the pathway for PCD thin film creation in [115] which will make better use of multiway interactions with dynamic complexation. We have also seen the use of multiway synchronisation including name passing in creating more detailed models of chemical reactions such as the Hydrogen Dibromide pathway of Example 4.3.3 – extending these isolated models of chemical bonding to a more systematic framework for formally modelling structural chemistry including, for example, allosteric modification of protein function, is an important challenge for the future.

How does our mechanism compare to multiway synchronisation in other process calculi? Bio-PEPA introduced the use of a multiway synchronisation to biochemical process algebras [33], and, as we saw in Section 5.2, this allows it to model many kinds of multiway interactions in biological systems. However, Bio-PEPA does not feature name passing or dynamic complexation, and its synchronisation mechanism is more similar to CSP style cooperation than π-calculus style communication, so we argue our mechanism is a better fit for a name passing calculus – the investigations in Chapter 3 support our view of our mechanism as a natural extension of symmetric name passing in the styles of the πI-calculus. A number of general process algebra including CSP [18] and E-LOTOS [64] also use a similar style of multiway synchronisation. Perhaps closer is the join-calculus which allows multiway synchronisation via pattern matching [61] – there is some similarity between join patterns in the join calculus and our use of affinity networks. Indeed, sites and locations are partially inspired by join patterns via other languages such as $\pi@$

[116] and SPiCO [78]. However, the join-calculus was developed for modelling distributed systems and we are not aware of it being applied in a quantitative biological context, so its primitives do not always have a clear biological interpretation (however, it does have a formal semantics in terms of the chemically inspired Reflexive Chemical Abstract Machine [60]), whereas, affinity networks and compatibility networks map more directly to existing ideas in biochemistry. It would be interesting to investigate further the extent to which join-patterns and compatibility networks can be related, and whether more ideas from the join-calculus can be brought over to a biological context.

There is also the link-calculus [12] which directly extends the π-calculus with a form of multiway synchronisation. In the link calculus multiway synchronisations are made up as chains of links; these allow names to combine to allow a multiway synchronisations in a way somewhat similar to the mπI calculus. Links may well be more flexible than name parts as they does not require the total length of the chain to be specified by each agent, however, we feel names parts or compatibility networks are a more natural model of chemical bonding as a link does not have a clear biological interpretation. The name passing mechanism is, however, significantly different, allowing each process to do a simultaneous send/receive with other communication partners. Multiway coordination is arguably simpler, and leads more readily to a direct biological interpretation. The link-calculus is a qualitative calculus, and it is unclear how quantitative rates would be associated to its multiway synchronisations. Nevertheless, we feel it may be worth further investigation of the relationship between these different mechanisms for multiway synchronisation.

General kinetic laws

In Chapter 5 we saw both general expressiveness results demonstrating our ability to express arbitrary autonomous differential equations, and in particular, chemical reaction networks. We should then, in theory, be able to support arbitrary kinetic laws[1]. We also considered a number of case studies applying our language applying general kinetics to real biological models, and saw this allowed us to effectively express a wide range of mathematical models covering biochemistry, ecological modelling, tumour/immune interactions, and gene regulation. As we discussed Section 5.2, these models could be

[1]With the possible exception of laws involving explicit time dependencies, which we do not expect to see in modelling chemical reactions, but might, however, appear in some population models or tumour treatment models. Our definition of kinetic laws could be extended to include a time variable, however, it is not clear how well this would within the framework of process algebra.

modelled similarly in other quantitative process algebra with support for general kinetics such as Bio-PEPA [33] and sCCP [13], however, however, could not be directly modelled[2] in process algebras based on the law of mass action such as continuous π and stochastic π.

As previously discussed, *compositionality of models* is one of the major goals driving the application of process algebra to biological modelling, meaning that models of a system's components can be combined to give a model the whole system. In developing our process semantics we had another slightly more subtle goal in mind: *compositionality of semantics*, meaning that the semantics of larger system can be built up using the semantics of their components. Compositionality is one of the main advantages of continuous π's vector field semantics, and this was one of the main reasons we developed a new vector field semantics, rather than building more directly on existing techniques for ODE extraction in process algebra with general kinetics support such as Bio-PEPA's fluid flow approximation [69, 33]. To what extent has our new semantics achieved this goal? The vector field semantics is compositional to at least some extend since we are able compose the semantics of mixtures as,

$$|\Pi|\Phi\rangle = |\Pi\rangle + |\Phi\rangle.$$

However, $|\Pi\rangle$ just records the potential of a process Π to engage in reactions; it is only when we compute $\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t}$ based on the global affinity networks that we will know what reactions will actually take place, determining the quantitative evolution of the system. As far as we can tell there is no way to build up $\frac{\mathrm{d}|\Pi\rangle}{\mathrm{d}t}$ in a compositional way (as continuous π does in the mass action case), given that general kinetic laws may cause the reaction velocity and direction to change in a non-linear way when new species are added to a mixture. Therefore, we have only been partially successful in giving a compositional semantics to general kinetics.

The process semantics we have developed is still substantially different from existing semantics for quantitative process algebras, and we believe that vector field semantics are worth investigating further as a step towards a denotational mathematical basis for interacting quantitative systems. It may be possible to relate our process semantics to the program of [49, 48, 95] who have investigated vector and operator semantics to Probabilistic Concurrent Constraint Programming.

[2]That is, without building the corresponding mass action model, and then finding parameter values for this model.

6.2 Future work

Implementation improvements

As part of this project a basic implementation of the language was developed. This does however currently have one main limitation: whilst direct simulation of models by applying numerical methods to the process semantics is supported, symbolic extraction of differential equations is not. This is more a limitation in the implementation than a theoretic issue, since the symbolic form of the differential equations can be derived directly from the process semantics as illustrated by examples in Section 4.6 and by the ODE extraction algorithm for continuous-π [80]. This currently limits simulation performance, since simulation must be performed on abstract process vectors, rather than using standard efficient ODE solvers. It is also worth investigating other potential performance improvements for process normal form computations, and, in particular, handling α-conversion, since this is a surprisingly challenging computational problem [76, 106].

Analysing topological dynamics of models

Whilst this dissertation has focused on new ways to formally express biological systems, a major motivation of building these formal model is to provide automated analysis of models. A substantial amount of work on automated analysis of continuous π models has already been carried out, with Banks and Stark's Logic of Behaviour in Context ($\mathcal{LBC}$) [8] providing a temporal logic and model checking techniques for to verify properties of models (for example, the concentration of a given species will fall below 0.001 and remain so indefinitely). This is able to make use of the compositional nature of models to verify properties such as if P is introduced to a context C at any point in the time interval $[0, 5]$ then species S will die out in interval $[10, 15]$, which depend on the interactions of P with a context process C.

This type of analysis performed by $\mathcal{LBC}$ focuses on the evolution of the process from a given starting point, and relies on inspecting traces from numerical solution of the differential equations. However, there is a long tradition of mathematical analysis of differential equations which looks at the structure of their whole phase space to understand their qualitative/topological dynamics. For example, we can find *invariant sets*, which no trajectory starting in may ever leave, and attractors to which any trajectory starting in their *basin of attraction* will be drawn. Future work could be carried out to use topolog-

ically dynamics to automatically analyse qualitative properties of models – for example, we can answer the question of whether a species S will die out if we find that $[S] = 0$ is an attracting set and the starting trajectory in within the basin of attraction. Traditionally, automated methods for verifying proving qualitative properties of dynamical systems relied on finding explicit solutions to the equations [81], which is not possible for many of the models we have considered, especially once we include general kinetics. However, recent methods including [83, 110] make it possible to find many invariant sets by whilst only looking at the differential equation, without the need for an explicit solution.

It may also be worth investigating whether the process algebra structure of models sheds any light on their topological dynamics. For the mass action case, it is known that the differential equations of the chemical reaction network are of a quite special form, and, for example, the stoichiometric matrix can reveal linear conservation laws, and there has been some research of finding non-linear conservation laws [2, 7]. However, it is unclear how much insight the network or process algebraic structure can give us about models using general kinetic laws. There is some existing work on verifying properties of Bio-PEPA models by considering both simulation traces and dynamical properties such as invariant and sources/sinks, which makes use of the model structure [38]. This has been applied to analysing conservation of mass in models [37].

Stochastic and hybrid simulation

The current work on continuous π and the continuous bond-calculus has been heavily focused on discrete state continuous simulations via differential equation extraction. However, a major advantage of process calculi is that the syntax of the model is separated from the semantics, allowing a single model to yield multiple different simulations. In future our language and semantics could be extended to allow stochastic simulation of models based on CTMCs and Gillespie's Stochastic Simulation Algorithm (SSA) in a manner similar to Bio-PEPA [33], Stochastic π [98], or sCCP [13]. This would us to perform stochastic simulation of all of the existing models we have considered. It may also be worth investigating simulating models as Stochastic Differential Equations (SDEs) using the Chemical Langevin Equation. This would allow us to take into account some of the stochastic effects whilst providing significantly better simulation performance than the full SSA. There has been some investigation of SDE approximations of PEPA models, although this method of simulation is not widely used in biological process algebras [109].

It also be interesting to consider modelling hybrid systems including for example,

discrete events or time delays considered by [16, 62] and [23]. This would be particularly interesting for extending the immune interaction model we developed in Section 5.2.3, since there has been interest in using hybrid events and time delays to give better immunological models [21].

Bibliography

[1] Continuous and spatial extension of stochastic π-calculus. Master's thesis, Imperial College London, 2009. Available at: `http://pubs.doc.ic.ac.uk/fluid-spatial-pi-calculus/`.

[2] Claudia Valls Carsten Wiuf Adam Mahdi, Antoni Ferragut. Conservation laws in biochemical reaction networks. Accessible at: `http://users.ox.ac.uk/~engs1301/papers/CRN_Darboux.pdf`.

[3] Ioan I. Ardelean and Matteo Cavaliere. Modelling biological processes by using a probabilistic p system software. *Natural Computing*, 2(2):173–197, Jun 2003.

[4] William Y. Arms. *The Early Years of Academic Computing: A Memoir by William Y. Arms*. 2014. Accessible at: https://ecommons.cornell.edu/handle/1813/36926.

[5] Giorgio Bacci, Davide Grohmann, and Marino Miculan. Bigraphical models for protein and membrane interactions. In *Proceedings Third Workshop on Membrane Computing and Biologically Inspired Process Calculi, MeCBIC 2009, Bologna, Italy, 5th September 2009.*, pages 3–18, 2009.

[6] Giorgio Bacci, Davide Grohmann, and Marino Miculan. A framework for protein and membrane interactions. In *Proceedings Third Workshop on Membrane Computing and Biologically Inspired Process Calculi, MeCBIC 2009, Bologna, Italy, 5th September 2009.*, pages 19–33, 2009.

[7] John C Baez and Jacob Biamonte. Quantum techniques for stochastic mechanics. *arXiv preprint arXiv:1209.3632*, 2012.

[8] C.J. Banks and I. Stark. A logic of behaviour in context. *Inf. Comput.*, 236(C):3–18, August 2014.

[9] Vladimir Batagelj and Andrej Mrvar. Some analyses of Erdős' collaboration graph. *Social Networks*, 22(2):173 – 186, 2000.

[10] Jan A Bergstra and Jan Willem Klop. Algebra of communicating processes. *CWI Monograph series*, 3:89–138, 1986.

[11] Ralf Blossey, Luca Cardelli, and Andrew Phillips. A compositional approach to the stochastic dynamics of gene networks. In *Transactions on Computational Systems Biology IV*, pages 99–122. Springer, 2006.

[12] Chiara Bodei, Linda Brodo, and Roberto Bruni. *Open Multiparty Interaction*, pages 1–23. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[13] Luca Bortolussi. Stochastic concurrent constraint programming. *Electronic Notes in Theoretical Computer Science*, 164(3):65 – 80, 2006. Proceedings of the 4th International Workshop on Quantitative Aspects of Programming Languages (QAPL 2006).

[14] Luca Bortolussi and Alberto Policriti. Stochastic concurrent constraint programming and differential equations. *Electronic Notes in Theoretical Computer Science*, 190(3):27 – 42, 2007. Proceedings of the Fifth Workshop on Quantitative Aspects of Programming Languages (QAPL 2007).

[15] Luca Bortolussi and Alberto Policriti. Dynamical systems and stochastic programming: To ordinary differential equations and back. In *Transactions on Computational Systems Biology XI*, pages 216–267. Springer, 2009.

[16] Luca Bortolussi and Alberto Policriti. Hybrid dynamics of stochastic programs. *Theoretical Computer Science*, 411(20):2052 – 2077, 2010. Hybrid Automata and Oscillatory Behaviour in Biological Systems.

[17] Luca Bortolussi and Alberto Policriti. Hybrid dynamics of stochastic programs. *Theor. Comput. Sci.*, 411(20):2052–2077, April 2010.

[18] Stephen D Brookes, Charles AR Hoare, and Andrew W Roscoe. A theory of communicating sequential processes. *Journal of the ACM (JACM)*, 31(3):560–599, 1984.

[19] Muffy Calder, Stephen Gilmore, and Jane Hillston. *Modelling the Influence of RKIP on the ERK Signalling Pathway Using the Stochastic Process Algebra PEPA*, pages 1–23. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[20] Igor Cappello and Paola Quaglia. A translation of beta-binders in a prioritized pi-calculus. *Electronic Notes in Theoretical Computer Science*, 229(1):109–125, 2009.

[21] Giulio Caravagna, Alex Graudenzi, Marco Antoniotti, Giancarlo Mauri, and Alberto d'Onofrio. Effects of delayed immune-response in tumor immune-system interplay. *arXiv preprint arXiv:1208.3855*, 2012.

[22] Giulio Caravagna and Jane Hillston. Modeling biological systems with delays in Bio-PEPA. In *MeCBIC*, pages 85–101, 2010.

[23] Giulio Caravagna and Jane Hillston. Bio-PEPAd: A non-Markovian extension of Bio-PEPA. *Theoretical Computer Science*, 419:26–49, 2012.

[24] Marco Carbone and Sergio Maffeis. On the expressive power of polyadic synchronisation in π-calculus. *Electronic Notes in Theoretical Computer Science*, 68(2):15 – 32, 2002. EXPRESS'02, 9th International Workshop on Expressiveness in Concurrency (Satellite Workshop of CONCUR 2002).

[25] Luca Cardelli. Brane calculi. In *International Conference on Computational Methods in Systems Biology*, pages 257–278. Springer, 2004.

[26] Luca Cardelli. Bitonal membrane systems interactions of biological membranes. *Theoretical Computer Science*, 404(1-2):5–18, 8 2008.

[27] Luca Cardelli. *Strand Algebras for DNA Computing*, pages 12–24. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[28] Luca Cardelli and Andrew D. Gordon. Mobile ambients. *Theoretical Computer Science*, 240(1):177 – 213, 2000.

[29] Alonzo Church. A set of postulates for the foundation of logic. *Annals of Mathematics*, 33(2):346–366, 1932.

[30] Federica Ciocchetta. *The BlenX Language with Biological Transactions*, pages 114–152. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[31] Federica Ciocchetta. Transactions on computational systems biology xi. chapter Bio-PEPA with Events, pages 45–68. Springer-Verlag, Berlin, Heidelberg, 2009.

[32] Federica Ciocchetta and Maria Luisa Guerriero. Modelling biological compartments in Bio-PEPA. *Electronic Notes in Theoretical Computer Science*, 227:77 – 95, 2009. Proceedings of the Second International Meeting on Membrane Computing and Biologically Inspired Process Calculi (MeCBIC 2008).

[33] Federica Ciocchetta and Jane Hillston. Process algebras in systems biology. *Formal methods for computational systems biology*, pages 265–312, 2008.

[34] Federica Ciocchetta and Jane Hillston. Bio-PEPA for epidemiological models. *Electronic Notes in Theoretical Computer Science*, 261:43 – 69, 2010. Proceedings of the Fourth International Workshop on the Practical Application of Stochastic Modelling (PASM 2009).

[35] Federica Ciocchetta and Corrado Priami. Beta-binders with biological transactions. Technical report, University of Trento, 2006.

[36] Federica Ciocchetta and Corrado Priami. Biological transactions for quantitative models. *Electronic Notes in Theoretical Computer Science*, 171(2):55 – 67, 2007. Proceedings of the First Workshop on Membrane Computing and Biologically Inspired Process Calculi (MeCBIC 2006).

[37] Allan Clark, Stephen Gilmore, Maria Luisa Guerriero, and Jane Hillston. Conservation of mass analysis for Bio-PEPA. *Electronic Notes in Theoretical Computer Science*, 296:107 – 126, 2013. Proceedings the Sixth International Workshop on the Practical Application of Stochastic Modelling (PASM) and the Eleventh International Workshop on Parallel and Distributed Methods in Verification (PDMC).

[38] Allan Clark, Stephen Gilmore, Maria Luisa Guerriero, and Peter Kemper. On verifying Bio-PEPA models. In *Proceedings of the 8th International Conference on Computational Methods in Systems Biology*, CMSB '10, pages 23–32, New York, NY, USA, 2010. ACM.

[39] M. Curti, P. Degano, C. Priami, and C.T. Baldari. Modelling biochemical pathways through enhanced ΪĂ-calculus. *Theoretical Computer Science*, 325(1):111 – 140, 2004. Computational Systems Biology.

[40] Michele Curti, Pierpaolo Degano, and Cosima Tatiana Baldari. *Causal π-Calculus for Biochemical Modelling*, pages 21–34. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[41] Troels C Damgaard, Vincent Danos, and Jean Krivine. A language for the cell. In *Technical Report TR-2008-116*. 2008.

[42] Vincent Danos, Jérôme Feret, Walter Fontana, Russell Harmer, Jonathan Hayman, Jean Krivine, Chris Thompson-Walsh, and Glynn Winskel. Graphs, rewriting and

pathway reconstruction for rule-based models. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 18. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2012.

[43] Vincent Danos and Jean Krivine. Formal molecular biology done in ccs-r. *Electronic Notes in Theoretical Computer Science*, 180(3):31 – 49, 2007. Proceedings of the First Workshop on Concurrent Models in Molecular Biology (BioConcur 2003).

[44] Vincent Danos and Cosimo Laneve. Formal molecular biology. *Theoretical Computer Science*, 325(1):69–110, 2004.

[45] R. David and H. Alla. *Discrete, Continuous, and Hybrid Petri Nets*. Springer Berlin Heidelberg, 2010.

[46] André M. de Roos. Modeling population dynamics. Lecture notes. Available at: `https://staff.fnwi.uva.nl/a.m.deroos/downloads/pdf_readers/syllabus.pdf`.

[47] L. Dematté, C. Priami, and A. Romanel. The BlenX language: A tutorial. In *Proceedings of the Formal Methods for the Design of Computer, Communication, and Software Systems 8th International Conference on Formal Methods for Computational Systems Biology*, SFM'08, pages 313–365, Berlin, Heidelberg, 2008. Springer-Verlag.

[48] Alessandra Di Pierro, Chris Hankin, and Herbert Wiklicky. Quantitative relations and approximate process equivalences. In *International Conference on Concurrency Theory*, pages 508–522. Springer, 2003.

[49] Alessandra Di Pierro and Herbert Wiklicky. Concurrent constraint programming: Towards probabilistic abstract interpretation. In *Proceedings of the 2Nd ACM SIGPLAN International Conference on Principles and Practice of Declarative Programming*, PPDP '00, pages 127–138, New York, NY, USA, 2000. ACM.

[50] Edsger W. Dijkstra. Sets are unibags, 1981. Accessible at: `https://www.cs.utexas.edu/users/EWD/transcriptions/EWD07xx/EWD786a.html`.

[51] Denys Duchier and Céline Kuttler. Biomolecular agents as multi-behavioural concurrent objects. *Electronic Notes in Theoretical Computer Science*, 150(1):31 – 49, 2006. Proceedings of the First International Workshop on Methods and Tools for Coordinating Concurrent, Distributed and Mobile Systems (MTCoord 2005).

[52] Xenia Efthymiou and Anna Philippou. A process calculus for spatially-explicit ecological models. *Application of Membrane Computing, Concurrency and Agent-based Modelling in Population Biology (AMCA-POP 2010)*, pages 84–78, 2010.

[53] Michael B. Elowitz and Stanislas Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–338, 2000.

[54] Heiko Enderling and Mark AJ Chaplain. Mathematical modeling of tumor growth and treatment. *Current pharmaceutical design*, 20(30):4934–4940, 2014.

[55] F Fages. Modelling and querying interaction networks in the biochemical abstract machine biocham. *Journal of Biological Physics and Chemistry*, 4(2):64–73, 2 2002.

[56] Sofia Farkona, Eleftherios P. Diamandis, and Ivan M. Blasutig. Cancer immunotherapy: the beginning of the end of cancer? *BMC Medicine*, 14(1):73, May 2016.

[57] Cheng Feng and Jane Hillston. *PALOMA: A Process Algebra for Located Markovian Agents*, pages 265–280. Springer International Publishing, Cham, 2014.

[58] Grazziela P. Figueredo, Peer-Olaf Siebers, Markus R. Owen, Jenna Reps, and Uwe Aickelin. Comparing stochastic differential equations and agent-based modelling and simulation for early-stage cancer. *PLOS ONE*, 9(4):1–18, 04 2014.

[59] Walter Fontana and Leo W. Buss. "the arrival of the fittest": Toward a theory of biological organization. *Bulletin of Mathematical Biology*, 56(1):1 – 64, 1994.

[60] Cédric Fournet and Georges Gonthier. The reflexive cham and the join-calculus. In *Proceedings of the 23rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 372–385. ACM, 1996.

[61] Cédric Fournet and Georges Gonthier. *The Join Calculus: A Language for Distributed Mobile Programming*, pages 268–332. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[62] Vashti Galpin, Luca Bortolussi, and Jane Hillston. HYPE: Hybrid Modelling by Composition of Flows. *Form. Asp. Comput.*, 25(4):503–541, July 2013.

[63] Xiaojing J Gao and Michael B Elowitz. Synthetic biology: precision timing in a cell. *Nature*, 538(7626):462–463, 2016.

[64] Hubert Garavel and Mihaela Sighireanu. Towards a second generation of formal description techniques–rationale for the design of E-LOTOS. In *Proceedings of the 3rd International Workshop on Formal Methods for Industrial Critical Systems FMICS*, volume 98, pages 187–230, 1998.

[65] Nil Geisweiller, Jane Hillston, and Marco Stenico. Relating continuous and discrete PEPA models of signalling pathways. *Theoretical Computer Science*, 404(1):97 – 111, 2008. Membrane Computing and Biologically Inspired Process Calculi.

[66] Peter JE Goss and Jean Peccoud. Quantitative modeling of stochastic systems in molecular biology by using stochastic petri nets. *Proceedings of the National Academy of Sciences*, 95(12):6750–6755, 1998.

[67] Jerrold W Grossman. Patterns of collaboration in mathematical research. *SIAM News*, 35(9):8–9, 2002.

[68] Jonathan Hayman and Tobias Heindel. Pattern graphs and rule-based models: The semantics of kappa. In *FoSSaCS*, volume 7794, pages 1–16. Springer, 2013.

[69] J. Hillston. Fluid flow approximation of PEPA models. In *Proceedings of the Second International Conference on the Quantitative Evaluation of Systems*, QEST '05, pages 33–, Washington, DC, USA, 2005. IEEE Computer Society.

[70] Jane Hillston. Compositional Markovian modelling using a process algebra. In *Computations with Markov chains*, pages 177–196. Springer, 1995.

[71] CAR Hoare. Communicating sequential processes. *Communications of the ACM*, 21(8):666–677, 1978.

[72] Ole Høgh Jensen and Robin Milner. Bigraphs and mobile processes (revised). Technical report, University of Cambridge, Computer Laboratory, 2004.

[73] Mathias John, Cédric Lhoussaine, and Joachim Niehren. Dynamic Compartments in the Imperative Pi Calculus. In *Computational Methods in Systems Biology, 7th International Conference*, volume 5688, pages 235–250, Bologna, Italy, August 2009. Spinger.

[74] Mathias John, Cédric Lhoussaine, Joachim Niehren, and Adelinde M. Uhrmacher. *The Attributed Pi Calculus*, pages 83–102. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[75] Mathias John, Cédric Lhoussaine, Joachim Niehren, and Adelinde M. Uhrmacher. *The Attributed Pi-Calculus with Priorities*, pages 13–76. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[76] V. Khomenko and R. Meyer. Checking pi-calculus structural congruence is graph isomorphism complete. In *2009 Ninth International Conference on Application of Concurrency to System Design*, pages 70–79, July 2009.

[77] Jean Krivine, Robin Milner, and Angelo Troina. Stochastic bigraphs. *Electronic Notes in Theoretical Computer Science*, 218:73 – 96, 2008. Proceedings of the 24th Conference on the Mathematical Foundations of Programming Semantics (MFPS XXIV).

[78] Céline Kuttler, Cédric Lhoussaine, and Joachim Niehren. A stochastic pi calculus for concurrent objects. *Ab*, 4545:232–246, 2007.

[79] Vladimir A Kuznetsov, Iliya A Makalkin, Mark A Taylor, and Alan S Perelson. Nonlinear dynamics of immunogenic tumors: parameter estimation and global bifurcation analysis. *Bulletin of mathematical biology*, 56(2):295–321, 1994.

[80] Marek Kwiatkowski and Ian Stark. The continuous π-calculus: A process algebra for biochemical modelling. In *Computational Methods in Systems Biology: Process of the Sixth International Conference CMSB 2008*, number 5307 in Lecture Notes in Computer Science, pages 103–122. Springer-Verlag, 2008.

[81] Gerardo Lafferriere, George J. Pappas, and Sergio Yovine. Symbolic reachability computation for families of linear vector fields. *Journal of Symbolic Computation*, 32(3):231 – 253, 2001.

[82] V. Leskovac. *Comprehensive Enzyme Kinetics*. Springer US, 2007.

[83] Jiang Liu, Naijun Zhan, and Hengjun Zhao. Computing semi-algebraic invariants for polynomial dynamical systems. In *Proceedings of the Ninth ACM International Conference on Embedded Software*, EMSOFT '11, pages 97–106, New York, NY, USA, 2011. ACM.

[84] Chris McCaig, Rachel Norman, and Carron Shankland. Process algebra models of population dynamics. *Lecture notes in computer science*, 5147:139–155, 2008.

[85] Robin Milner. A calculus of communicating systems. 1980.

[86] Robin Milner. *Functions as processes*, pages 167–180. Springer Berlin Heidelberg, Berlin, Heidelberg, 1990.

[87] Robin Milner. *Communicating and mobile systems: the π-calculus*. Cambridge university press, 1999.

[88] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, i. *Information and Computation*, 100(1):1–40, 1992.

[89] M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409, 2001.

[90] Rachel Norman and Carron Shankland. Developing the use of process algebra in the derivation and analysis of mathematical models of infectious disease. *Lecture notes in computer science*, pages 404–414, 2003.

[91] Michael Pedersen and Gordon D. Plotkin. Transactions on computational systems biology xii. chapter A Language for Biochemical Systems: Design and Formal Specification, pages 77–145. Springer-Verlag, Berlin, Heidelberg, 2010.

[92] James L Peterson. Petri net theory and the modeling of systems. 1981.

[93] Andrew Phillips. A visual process calculus for biology. *Symbolic Systems Biology: Theory and Methods, Jones and Bartlett Publishers (to appear, 2010), http://research. microsoft. com/en-us/projects/spim*, 2009.

[94] Luca Phillips, Andrew Cardelli, Giuseppe Castagna, and Giuseppe Castagna. A graphical representation for biological processes in the stochastic pi-calculus. *Lecture Notes in Computer Science*, 4230:123–152, 2006.

[95] Alessandra Di Pierro and Herbert Wiklicky. Operator algebras and the operational semantics of probabilistic languages. *Electronic Notes in Theoretical Computer Science*, 161:131 – 150, 2006. Proceedings of the Third Irish Conference on the Mathematical Foundations of Computer Science and Information Technology (MFCSIT 2004).

[96] Gordon D Plotkin. A structural approach to operational semantics. 1981.

[97] Laurent Potvin-Trottier, Nathan D Lord, Glenn Vinnicombe, and Johan Paulsson. Synchronous long-term oscillations in a synthetic gene circuit. *Nature*, 538(7626):514–517, 2016.

[98] C. Priami. Stochastic π-calculus. *The Computer Journal*, 38(7):578–589, 1995.

[99] Corrado Priami and Paola Quaglia. *Beta Binders for Biological Interactions*, pages 20–33. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[100] P. Prusinkiewicz, J.S. Hanan, F.D. Fracchia, A. Lindenmayer, D.R. Fowler, M.J.M. de Boer, and L. Mercer. *The Algorithmic Beauty of Plants*. The Virtual Laboratory. Springer New York, 2012.

[101] G. Păun. *Computing with Membranes*. TUCS technical report. Turku Centre for Computer Science, 1998.

[102] Aviv Regev, Ekaterina M Panina, William Silverman, Luca Cardelli, and Ehud Shapiro. Bioambients: an abstraction for biological compartments. *Theoretical Computer Science*, 325(1):141–167, 4 2004.

[103] Aviv Regev, William Silverman, and Ehud Shapiro. Representation and simulation of biochemical processes using the π-calculus process algebra. In *Pacific symposium on biocomputing*, volume 6, pages 459–470, 2001.

[104] Wang S. Modelling biological systems as communicating processes. Masters dissertation, University of Edinburgh, Supervised by Ian Stark.

[105] Davide Sangiorgi. π-calculus, internal mobility, and agent-passing calculi. *Theoretical Computer Science*, 167(1-2):235–274, 1996.

[106] Manfred Schmidt-Schauß, Conrad Rau, and David Sabel. Algorithms for extended alpha-equivalence and complexity. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 21. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.

[107] Ankit Sharma, Jaideep Srivastava, and Abhishek Chandra. Predicting multi-actor collaborations using hypergraphs. *arXiv preprint arXiv:1401.6404*, 2014.

[108] Padmanee Sharma and James P Allison. Immune checkpoint targeting in cancer therapy: toward combination strategies with curative potential. *Cell*, 161(2):205–214, 2015.

[109] Joris Slegers. A Langevin interpretation of PEPA models. *Electronic Notes in Theoretical Computer Science*, 261:71 – 89, 2010. Proceedings of the Fourth International Workshop on the Practical Application of Stochastic Modelling (PASM 2009).

[110] Andrew Sogokon. *Direct methods for deductive verification of temporal properties in continuous dynamical systems*. PhD thesis, University of Edinburgh, 2015.

[111] Carla Taramasco, Jean-Philippe Cointet, and Camille Roth. Academic team formation as evolving hypergraphs. *Scientometrics*, 85(3):721–740, Dec 2010.

[112] René Thomas. Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, 42(3):563 – 585, 1973.

[113] Suzanne L Topalian, George J Weiner, and Drew M Pardoll. Cancer immunotherapy comes of age. *Journal of Clinical Oncology*, 29(36):4828–4836, 2011.

[114] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, 1937.

[115] Paravee Vas-Umnuay, Ki-Joong Kim, and Chih-Hung Chang. Growth kinetics of copper sulfide thin films by photochemical deposition. *CrystEngComm*, 18(35):6748–6758, 2016.

[116] Cristian Versari and Roberto Gorrieri. *π@: A π-Based Process Calculus for the Implementation of Compartmentalised Bio-inspired Calculi*, pages 449–506. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[117] Ludovica Luisa Vissat, Jane Hillston, Glenn Marion, and Matthew J Smith. MELA: Modelling in Ecology with Location Attributes. *arXiv preprint arXiv:1610.08171*, 2016.

[118] Philip Wadler. A taste of linear logic. *Mathematical Foundations of Computer Science 1993*, pages 185–210, 1993.

[119] Chin-Rang Yang, Bruce E Shapiro, Eric D Mjolsness, and G Wesley Hatfield. An enzyme mechanism language for the mathematical modeling of metabolic pathways. *Bioinformatics*, 21(6):774–780, 2004.

[120] Roberto Zunino, Durica Nikolić, Corrado Priami, Ozan Kahramanoğulları, and Tommaso Schiavinotto. *ℓ: An Imperative DSL to Stochastically Simulate Biological Systems*, pages 354–374. Springer International Publishing, Cham, 2015.