An executable stochastic model for Fatty Acid metabolism

Argyris Zardilis

Department of Applied Mathematics and Theoretical Physics

University of Cambridge

This dissertation is submitted for the degree of

Master of Philosophy

St Catharine's College August 2014

I would like to dedicate this thesis to my loving parents ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration, except where specifically indicated in the text. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Argyris Zardilis
August 2014

Acknowledgements

And I would like to acknowledge ...

Abstract

This is where you write your abstract ...

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Relatively recent technological advances which led to an accumulation of a wealth of data that have made Biology as a discipline shift some of its efforts from understanding individual components to understanding systems of interacting components. A systems level understanding of biological system is really important as it is the distributed information processing that happens at the systems level that gives rise to phenotype and the macroscopic behaviour of cells to sustain life. Biology has not traditionally used any formal methods but as the systems under investigation grew in size some more formal approaches were needed. System Biology is a relatively new field that tries to fill the gap by bringing practises from traditionally more formal disciplines like Mathematics, Physics, and Computer Science into the study of biological systems.

In the area of metabolism we have gone from a study of the properties of single enzymes and metabolites to the study of the behaviour of entire metabolic pathways and even entire genome-wide metabolisms of mammalian and model organisms. This accumulated knowledge about the interconnections in and between metabolic pathways is deposited in large online databases that integrate informations from various sources. While these diagrams are a rich source of information since metabolism is a highly dynamic process sometimes a more dynamic picture is needed. Mathematical techniques like Flux Balance analysis originating from Dynamical Systems Theory are particularly popular in the analysis of the flows in and out of metabolic pathways.

We have recently been able to pinpoint the biochemical reasons behind different diseases in the loss of balance between anabolic and catabolic activities in cells. These activities are known to be tightly regulated to avoid excess production and accumulation of metabolites which can be the reasons behind observed disorders. The products and intermediaries of lipid metabolism are central in many metabolic disorders and diseases. While Flux Balance Analysis is of great importance in the study of large metabolic networks it is not so suited in studying more local metabolic processes and their crucial regulatory mechanisms.

Here we propose an alternative reaction-centric view of metabolic systems and lipid metabolism in particular which we think will be an important tool in the mechanistic characterisation of crucial local metabolic processes and their regulatory mechanisms. We also propose the formal language of Petri Nets, which has been successfully used before in biochemical networks, as a potential tool to capture this alternative reaction-centric view of lipid metabolism. The goals and the work done towards this project were therefore two-fold: Firstly create a reaction-centric model of the Fatty Acid(FA) synthesis/elongation process which is an important part of lipid metabolism and secondly assess the formal modelling language of Petri Nets as a potential tool to capture this view. Since part of the work done was about modelling methodology and since we noticed a more general trend towards executable formal models in biology we also present here an alternative version of the model in the process algebra of pi-calculus in an attempt to contrast the two formal modeling approaches: net models and process algebra models.

In this introduction I start by introducing the currently most popular methodology for the description of metabolic pathways(FBA), outline the biological importance of lipid metabolism and the motivation for a reaction-centric view, and finally I try to place Petri Nets and pi-calculus(the two methodologies used) in the formal distributed computing modelling language world.

1.1 Dynamical systems theory and Flux Balance Analysis

Systems Biology attempts to master the complexity and understand metabolic pathways and any biological system by usually contructing models. There are two types of models based on their construction method. Bottom-up models are constructed by extracting biological knowledge from experimental datasets. This category includes for example the diagrammatic means of capturing the interconnected structure of metabolic pathwayw. Top-down models on the other hand start from a detailed knowledge of the system to get a mechanistic model of the system.

Early attemps to capture the interactions and dependencies between components were made with bottom-up models which resulted in large diagrams that capture the conversion processes happending inside metabolic pathways. While these diagrams are important and are a useful represntation of our knowledge about metabolism they lack the dynamic nature which is crucial in a systems level understanding of a system which should not only include the connections between the components but also how these components interact over time, how their behaviour is controlled, and how they respond to external stimuli.

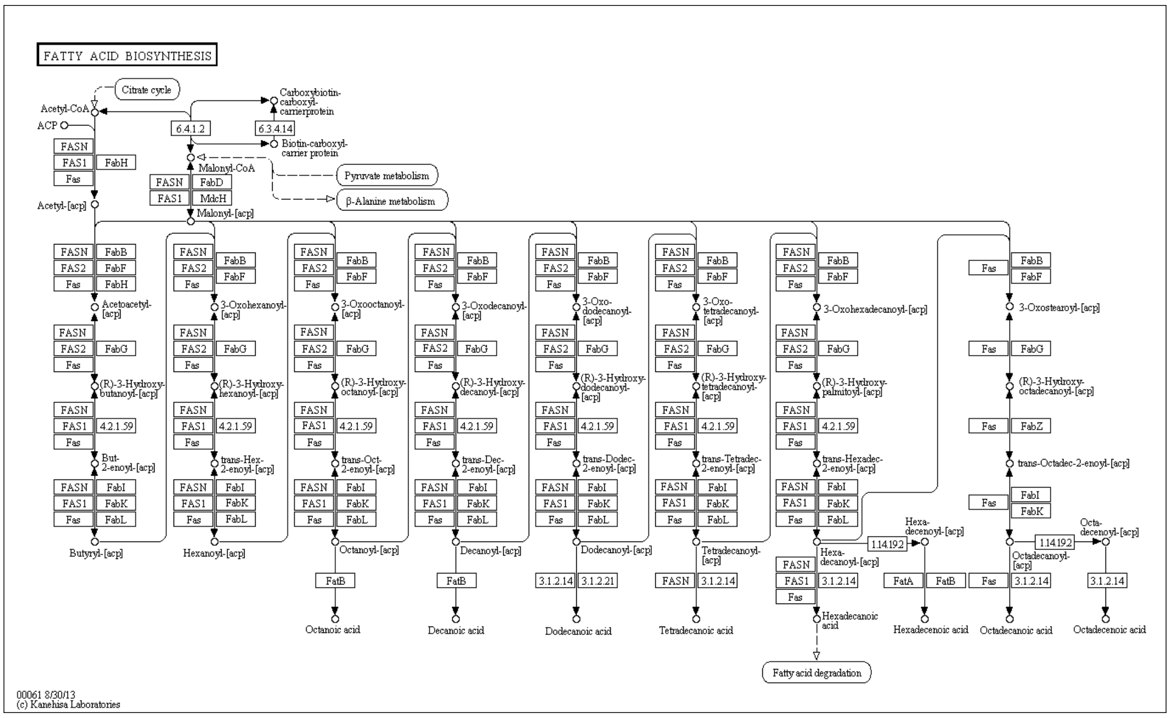


Fig. 1.1 Fatty acid synthesis pathway captured by the standard diagrammatic language used to capture interactions between components in a biological system. This was taken from KEGG a database which contains current knowledge about the working of these systems(from experiments for example) integrated with data from multiple sources.

This need to look into dynamic behaviour led to Dynamical System theory which based on its success in Physics has found its way in many other fields. Dynamical system theory captures the relationship between continuous quantities in difference-differential equations which describe the evolution of state variables in terms of changes in other state variables and even themselves thus capturing the interaction element. Since the time of Newton and classical mechanics ,where these ideas originated, the toolbox of dynamical systems theory has grown to include techniques for qualitative understanding of system without the need to solve them either numerically or analytically.

In biochemistry each species in the system is represented by its concentration leading to a differential equation for each species. The rate of change of the concentration of a species is described in terms of in flows- increasing the rate of change(production)- and out flows - decreasing the rate of change(degradation). These flows can be dependent on the concentrations of other variables(species) which participate in the same biochemical reactions. Consider for example a system with n components which we can group into the global state of the system $\mathbf{x} = (x_1, x_2, \dots x_n)$. All the reactions that take place in the system change, in discrete levels, the numbers of molecules of the species:

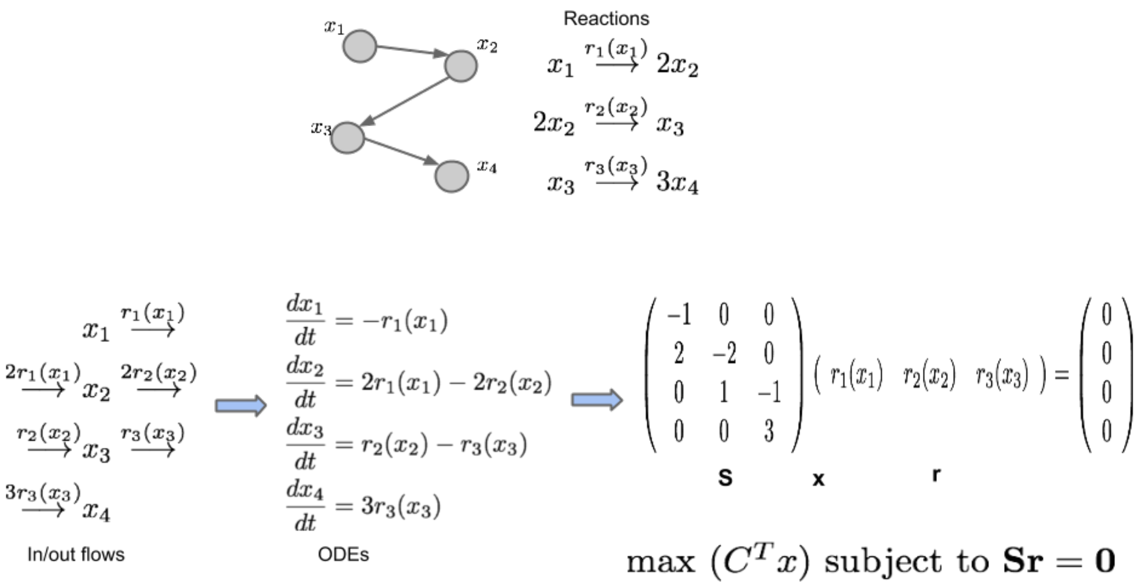
$$\begin{gathered}\mathbf{x} \xrightarrow{r_1(\mathbf{x})} \mathbf{x} + \delta_\mathbf{1} \\ \mathbf{x} \xrightarrow{r_2(\mathbf{x})} \mathbf{x} + \delta_\mathbf{2} \\ \vdots \\ \mathbf{x} \xrightarrow{r_m(\mathbf{x})} \mathbf{x} + \delta_\mathbf{m}\end{gathered}$$

So for every reaction we have one such rule and δ_k are the discrete levels by which the species change for every reaction. Each reaction has an associated and possible state dependent rate $r_k(x)$ which is the expected number of times it takes place in a time unit. The in-flows are the positive deltas and out-flows the negative ones. The differential equation describing the evolution of the average numbers of species i is then the sum of the in and out flows over all reactions in the system:

$$\frac{dx_i}{dt} = \sum_m r_m(\mathbf{x})\delta_{mi}$$

Usually this formulation of the problem leads to integration with a numerical ODE solver of the above system to get the dynamic behaviour of the system. The usual problem mathematical biologists face is that the reaction rate function(the $r_k(\mathbf{x})$s) contain parameter constants that are not usually known.

In the contex of metabolic systems however there is a mathematical technique however, called Flux Balance Analysis, that is used to compute these flows without explicit knowledge of these parameters by making some assumptions about the system to simplify the problem. If we assume that that metabolites cannot accumulate within the cell and their levels are more or less balanced then the system is at steady-state. The steady-state assumption means that the average positive flux-defined as the sum of all the in-flows- and the average negative flux-defined as the sum of all the out-flows- are equal which in turn means that the rate of change for every species , defined as the sum of all flows(negative and positive), is equal to 0. Solving for the fluxes leads to a linear equation for each species. For a more concise notation all the δ_{nm} are packaged into a matrix $\mathbf{S}$, traditionally called the Stoichiometric matrix, and the fluxes in a vector $\mathbf{r}$. Then the problem becomes solving the system $\mathbf{Sr} = \mathbf{0}$ to calculate the fluxes $\mathbf{r}$ through our system of interest. The only problem with this formulation of the problem is that the system is underdetermined since usually the number of reactions is higher than the number of species. To solve this the solution space is constrained with the use of an objective function which leads to a linear programming problem formulation which is easily solvable. An small contrived example to illustrate the above process is given in Figure 1.2.



x_1
x_2
x_3
x_4
Reactions
$x_1 \xrightarrow{r_1(x_1)} 2x_2$
$2x_2 \xrightarrow{r_2(x_2)} x_3$
$x_3 \xrightarrow{r_3(x_3)} 3x_4$
$x_1 \xrightarrow{r_1(x_1)}$
$\xrightarrow{2r_1(x_1)} x_2 \xrightarrow{2r_2(x_2)}$
$\xrightarrow{r_2(x_2)} x_3 \xrightarrow{r_3(x_3)}$
$\xrightarrow{3r_3(x_3)} x_4$
In/out flows
$\frac{dx_1}{dt} = -r_1(x_1)$
$\frac{dx_2}{dt} = 2r_1(x_1) - 2r_2(x_2)$
$\frac{dx_3}{dt} = r_2(x_2) - r_3(x_3)$
$\frac{dx_4}{dt} = 3r_3(x_3)$
ODEs
$\begin{pmatrix} -1 & 0 & 0 \\ 2 & -2 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 3 \end{pmatrix} \begin{pmatrix} r_1(x_1) & r_2(x_2) & r_3(x_3) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$
S
x
r
$\max\ (C^T x)$ subject to $\mathbf{Sr} = \mathbf{0}$

Fig. 1.2 A small example how starting from a bottom-up model like a pathway from KEGG we can go on to apply FBA and calculate the fluxes through our pathway

Flux balance analysis is a very powerful technique since it allows us to overcome the problem of the rate function parameters and calculate the fluxes through the system in a computationally efficient way. It is not without its limitations though.

1.2 Challenge of lipid metabolism

Lipid metabolism is particularly important topic since it is linked with several disorders and diseases. The fact that it is has complicated regulatory mechanisms has made getting a systems level understaning of it very important. In fact many system biology attempts have been made to get even a genome-wide picture of metabolism in mammals or other organisms. We think that lipid metabolism Systems Biology modelling might benefit from a different type of approach than the deterministic picture offered by dynamic systems theory and consequently FBA. The iterative nature of many of its subsystems, like the Fatty Acid Synthesis/Elongation system which we examine in this study, along with the need to understand more local regulatory mechanims calls for a reaction-centric view of the system.

A biochemical system, like the one given in the previous section, consisting of a number of reactions that alter the integer numbers of molecules of the species describes a random process and a Continuous Time Markov Chain in particular. The differential equations we used as a basis for FBA are a just deterministic view of the average behaviour. The dynamics of the system can be set in motion inside a computer using the Gillespie exact simulation algorithm which creates sample time-series. Statistical information like the strenth of the fluctuations (normalised variance) can be computed by collecting a lot of these traces.

In this study we propose Petri Nets as an alternative language to capture this reaction-centric view which we think is more expressive than the standard Markov processes formulation. While Petri Nets and their evolution through their formal operational semantics also described a CTMC and are thus equivalent in power with the standard Markov jump processes we believe that they posses some characteristics(see next section and Methods) that makes them useful in the analysis of the reaction-centric view of lipid metabolism. Language is important not only because a syntax is needed to define our models in but also because the notation we use is our tool of thought and the use of the correct language for thinking about a problem can enhance our understadnding and ultimately help us solving it. That is why we have more than one high-level programming language despite the fact that all of them are Turing complete and therefore have the ability to express all computations.

In the next section we try to place Petri Nets, the main modelling language used, and pi-calculus, the alternative used as a way to contrast net and process algebras models, in the spectra of formal modelling methodologies for distributed computation.

1.3 Computational models in Biology

Computer Science, although a fundamentally different discipline than Biology, has undergone a similar transformation in its focus from single information processing entities to systems of interacting entities(see the Internet, clusters). The term computation at the distributed level is now broader; it not only describes mere calculation but it also includes the interaction between components. This change in the term computation can be seen through the models we use to capture the notion of computation. At the early days of computing(even before actual electronic computers) the models of computation included lambda-calculus and Turing Machines. These early formalisms capture computation differently, Turing Machine as global state transitions in an Abstract Machine and lambda-calculus as reduction rules in a calculus, but they are effectively equivalent in expressive power(see Church-Turing thesis). As computer systems grew there was an interest for similar minimalistic languages to express distributed computation. Some formalisms that were invented during that period were Milner [7] Calculus of Communicating Systems, Lafont [6] Interaction nets, Milner et al. [8] pi-calculus, and Petri Nets(Murata [9]).

The analogy between distributed computer systems and biochemical systems was made explicit by Berry and Boudol [1] and the Chemical Abstract Machine. The modelling langugages used for distributed computation are applicalbe in biological systems since they both a high number of concurrent components with dependencies introducing non-determinism. Fontana and Buss [5] went as far as to propose pi-calculus as an alternative to dynamical systems theory since he considered a calculus of objects that can change and their interactions more appropriate for describing biochemical systems than a calculus of continuous interacting quantities. Priami et al. [10] went on to use a stochastic version of pi-calculus to describe a biochemical systems and from then on other Computer Science inspired formalisms came up like Cardelli [2] with Brane-calculi, Danos and Laneve [4] with kappa, and Ciocchetta and Hillston [3] with Bio-PEPA.

Petri Nets lean towards the automata and Turing Machines style of computation because their operational semantics are in terms of state transitions from a distributed global state. They have a very intuitive graphical notation and concurrency, non-determinism, and the causal independencies between events is inherent in their structure. They also have a very natural correspondence to biochemical reactions and in their basic form have been used to capture qualitative properties of biochemical systems. In this study we use the stochastic version of Petri Nets which explicitly models time to describe and quantitatively analyse the reaction-centric view of the FA elongation and synthesis process.

Process algebras, which pi-calculus is an example of, on the other hand define syntax to specify the behaviour and state transitions of the individual components of the system independently therefore reactions and compositionality are not explicitly captured. In this study I particularly use Stochastic Pi Machine (SPiM) which is a programming language derived from stochastic pi-calculus but extended to include operational semantics for execution on an Abstract Machine and also a formal graphical notation. The fact that this variant shares characteristics from both net model and process algebras approached makes it an interesting case-study.

1.4 Outline of work

To summarise the main aim of this study was to assess the applicability of the Petri net formal language to capture a reaction-centric view of lipid metabolism by producing a model of the Fatty Acid biosyntesis/elongation process. For completeness we also created a stochastic pi-calculus version of the model to contrast the two methodologies as part of the more general turn towards executable formal models in Biology. An extended version of the basic process is also given including part of its regulatory mechanisms.

In chapter 2 an overview of the methods used in modelling the process is given, namely Petri Nets and stochastic pi-calculus. In section 3 the basic and extended model are presented along with their tuning with available metabolomics data. Finally in section **??**

Chapter 2

Methods

2.1 Data sources

2.2 Petri Nets

2.2.1 Syntax and Semantics of Basic nets

A Petri Net is a 4-tuple $(P, T, pre, post)$ defined as:

- a set of *places* (or conditions) P

- a set of *transitions* (or events) T

- a *preconditions map pre* : $T \rightarrow \mathbf{m}P$ which assigns a multiset of places $pre(t)$ to each transition $t \in T$

- a *postconditions map post* : $T \rightarrow \mathbf{m}P$ which assigns a multiset of places $post(t)$ to each transition $t \in T$

where $\mathbf{m}P$ is the space of multisets over P with a multiset over P defined as function $f : P \rightarrow \mathbb{N}$. The state of the system, called a marking, is again a multiset $\mathscr{M}$ over P. We can think of the marking as the distributed global state of the system. It is also common when defining a Petri Net to give the initial marking of the system usually written as $\mathscr{M}_0$.

The operational semantics of Petri Nets is defined in terms of changes in the global state through the action of the transitions T of the PN:

$$\mathscr{M} \xrightarrow{t} \mathscr{M}'$$

Here when a transition(event) t occurs it changes the state of the system from marking $\mathscr{M}$ to marking $\mathscr{M}'$. Unlike automata and Turing Machines however a transition does not occur from a single global state but instead it only affects part of the state:

$$\mathscr{M} \xrightarrow{t} \mathscr{M}' \text{ iff } pre(t) \leq \mathscr{M} \text{ and } \mathscr{M}' = \mathscr{M} - pre(t) + post(t)$$

For 2 multisets f and g over set X, $f \leq g$ is defined as $f \leq g \iff \forall x \in X f_x \leq g_x$. So a transition t is said to be 'enabled' if its preconditions are sufficiently marked($pre(t) \leq \mathscr{M}$) and when it 'fires' it changes the marking in the places in its vicinity, namely the set of places $\{p \mid pre(t)_p \neq 0 \text{ or } post(t)_p \neq 0\}$($\mathscr{M}' = \mathscr{M} - pre(t) + post(t)$). The *post* and *pre* condition maps define the causal independence between transitions and add the ability of Petri Nets to model concurrency and non-determinism. Two (or more) transitions are concurrent if they do not share any preconditions. Non-determinism is added through dependencies of transitions which share preconditions. In that case a race condition is created because the dependent transitions compete at their shared preconditions with only one of them being able to fire at each step.

This formal definition of transitions leads to an algorithm for the executions of a PN as follows:

1. Initialise the net with $(P, T, pre, post)$ and set state $\mathscr{M} = \mathscr{M}_0$
2. Find enabled transitions, $enabled \subseteq T, \forall t \in T$ if $pre(t) \leq M$
3. Choose transition t from set of enabled at random
4. Update state according to $\mathscr{M} = \mathscr{M} - pre(t) + post(t)$
5. Repeat steps 2-4 until there are no more enabled transitions or an external stop condition is met(e.g. max number of steps)

The main strength of Petri Nets though lies on their very intuitive and widely used graphical representation. Petri Nets are represented as bipartite graphs with two sets of nodes, the places P and transitions T which are denoted by circles and rectangles respectively. A weighted directed edge with weight $n > 0$ is added between place p and transition t in the direction $p \to t$ if $pre(t)_p = n$. A weighted directed edge with weight $n > 0$ is added between transition t and place p in the direction $t \to p$ if $post(t)_p = n$. The markings are denoted as dots (tokens) in the circles depicting the places. So for example consider the net with $P = \{p1, p2, p3\}$, $T = \{t1\}$, $pre = \{(t1, \{(p1, 2), (p2, 1), (p3, 0)\})\}$, $post = \{(t1, \{(p1, 0), (p2, 0), (p3, 2)\})\}$, and marking $\{(p1, 3), (p2, 1), (p3, 1)\}$. The graphical depiction of this net would be the one shown in Figure 2.1. We could then also define the pre-places of a transition t as all the places p such that $pre(t)_p > 0$ and the post-places as all the places p such that $post(t)_p > 0$.

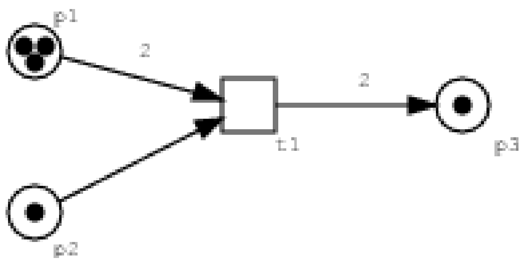


Fig. 2.1 Small basic Petri Net example

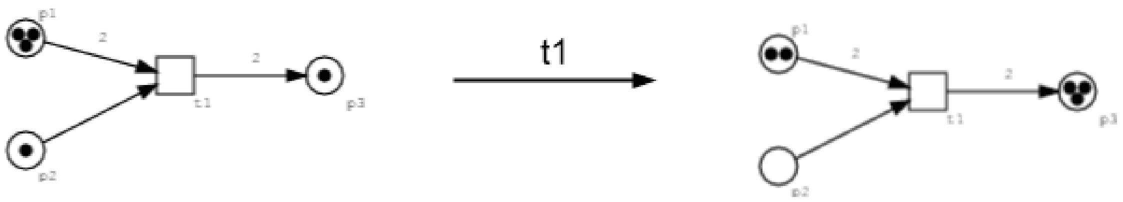


Fig. 2.2 The token game for basic Petri Nets.

The operational semantics of Petri Nets can also be defined very naturally in this graphical notation. When a transition t fires $pre(t)_p$ tokens are consumed from all the pre-places p and $post(t)_p$ tokens are added to all the post-places p. For the net given above(Figure 2.1) when transition $t1$ fires the 2 tokens are removed from $p1$ and 1 from $p2$ and 2 are added to $p3$(see Figure 2.2).The execution of the net can be seen as the flow of tokens through the net as transitions fire at each step according to the execution algorithm given above. This graphical view of the execution is called the 'token game' and it is very useful for gaining an insight into the dynamic behaviour of the system being modelled.

2.2.2 Stochastic Petri Net extension

Stochastic Petri Nets are an extension to the original Petri Net formalism to explicitly model time. Notice that even though the there is an ordering of transitions implied by their execution according to the algorithm given in the previous section the concept of time is not modelled explicitly in the basic Petri Net formalism. In order to model time explicitly Stochastic Petri Nets (SPNs) introduce a waiting time associated with each transition t, defined as random variable X_t distributed exponentially with potentially marking-dependent rate $\lambda_t(pre\_places(t))$ where pre_places is defined as before. Formally a SPN is a 5-tuple

$(P, T, pre, post, \nu)$ with everything defined as before with the addition of the map $\nu : T \to H$ where H is the set of all hazard functions $H = \{h_t \mid h_t : \mathbb{N}^{|pre(t)|} \to \mathbb{R}\}$. Hazard function is the typical name used in stochastic processes for the function giving the rate of the exponential time variable. In this case the hazard function h_t gives λ_t and the domain of h_t we will restrict to only the marking of the pre-places of t. This wait time associated with each transition changes the execution algorithm for Petri Nets; now when more than one transition is enabled a wait time is sampled from all of them and the one with the least waiting time gets to fire:

1. Initialise the net with $(P, T, pre, post, \nu)$, set state $\mathscr{M} = \mathscr{M}_0$, and $time = 0$
2. Find enabled transitions, $enabled \subseteq T, \forall t \in T$ if $pre(t) \leq M$
3. For each enabled transition t sample a wait time from an exponential distribution with rate $\lambda_t(pre\_places(t))$. Pick the transition t_i with least wait time to fire.
4. Proceed time $time = time + \tau_i$
5. Update state according to $\mathscr{M} = \mathscr{M} - pre(t_i) + post(t_i)$
6. Repeat steps 2-4 until there are no more enabled transitions or an external stop condition is met(e.g. max number of steps)

With this new variant of Petri Nets we maintain the non-determinism through the competition of enabled transitions with shared preconditions but by changing their rates we can favour some transitions over others. This Stochastic Petri Net formulation gives rise to Continuous Time Markov Chain and the execution algorithm defined above is more or less the Gillespie algorithm which is an exact algorithm for simulation of Markov jump processes. It is this Petri Net variant that we will use to model our system of interest, the Fatty Acid synthesis/elongation pathway. In the next section we describe the correspondence between a biochemical network and especially a metabolic pathway to a Stochastic Petri Net and some examples of previous uses of SPNs in biology/biochemistry.

2.2.3 Application to Biochemistry

The correspondence between a biochemical system and a Stochastic Petri Net is as follows: the chemical species of the system become places, reactions become transitions and their rates the rates of the transitions, and the stoichiometry of the reactions is represented in the *pre* and *post* maps. This is a very natural correspondence between chemical reactions and Petri Nets. In fact the standard notation for chemical notation is itself a process algebra albeit not a very formal one. So for example consider a made-up reaction that takes 2 molecules of A and 3 molecules of B and produces 5 molecules of C:

$$A + B \xrightarrow{r_1} 2C$$

This is an event, a reaction with rate k, which tells us how the state of the system changes but although implicitly we know the conditions for it to take place they are not formally captured by the informal process algebra of chemical reactions as defined by the standard arrow notation. A Petri Net representation of this reaction however captures the reaction more formally:

2.3 Stochastic Pi-calculus

This is going to contain a vey brief introduction to stochasic pi-calculus.

Chapter 3

Work

3.1 Basic model of Fatty Acid synthesis

3.1.1 Original and reduced order model

3.1.2 Point estimation of parameters

3.1.3 Exact likelihood and parameter profiles

3.1.4 An estimation of input flow parameters

3.1.5 A stochastic π-calculus version

3.2 Extended model of FA synthesis

Chapter 4

Discussion

References

[1] Berry, G. and Boudol, G. (1989). The chemical abstract machine. In *Proceedings of the 17th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 81–94. ACM.

[2] Cardelli, L. (2005). Brane calculi. In *Computational methods in systems biology*, pages 257–278. Springer.

[3] Ciocchetta, F. and Hillston, J. (2009). Bio-pepa: A framework for the modelling and analysis of biological systems. *Theoretical Computer Science*, 410(33):3065–3084.

[4] Danos, V. and Laneve, C. (2004). Formal molecular biology. *Theoretical Computer Science*, 325(1):69–110.

[5] Fontana, W. and Buss, L. W. (1996). *The barrier of objects: From dynamical systems to bounded organizations*. Citeseer.

[6] Lafont, Y. (1989). Interaction nets. In *Proceedings of the 17th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 95–108. ACM.

[7] Milner, R. (1980). A calculus of communicating systems.

[8] Milner, R., Parrow, J., and Walker, D. (1992). A calculus of mobile processes, ii. *Information and computation*, 100(1):41–77.

[9] Murata, T. (1989). Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580.

[10] Priami, C., Regev, A., Shapiro, E., and Silverman, W. (2001). Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information processing letters*, 80(1):25–31.

Appendix A

Installing the CUED class file

LaTeX.cls files can be accessed system-wide when they are placed in the <texmf>/tex/latex directory, where <texmf> is the root directory of the user's TeXinstallation. On systems that have a local texmf tree (<texmflocal>), which may be named "texmf-local" or "localtexmf", it may be advisable to install packages in <texmflocal>, rather than <texmf> as the contents of the former, unlike that of the latter, are preserved after the LaTeXsystem is reinstalled and/or upgraded.

It is recommended that the user create a subdirectory <texmf>/tex/latex/CUED for all CUED related LaTeXclass and package files. On some LaTeXsystems, the directory look-up tables will need to be refreshed after making additions or deletions to the system files. For TeXLive systems this is accomplished via executing "texhash" as root. MIKTeXusers can run "initexmf -u" to accomplish the same thing.

Users not willing or able to install the files system-wide can install them in their personal directories, but will then have to provide the path (full or relative) in addition to the filename when referring to them in LaTeX.