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 July 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
July 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

A systems level understanding of biology has always been in the back of the minds of biologists since it is this distributed information processing that ultimately gives rise to phenotype and macroscopic behaviour to sustain life. Perhaps the apparent great complexity at which these systems operate or the lack of relevant experimental data has made Biology as a discipline to focus its efforts on the understanding of the information processing capabilities and inner working of the individual constituent components of these systems. Relatively recent technological advances however have led to an accumulation of a wealth of data sheding light into the interactions between components, their dependencies and the constraints these impose. While understanding the individual components, genes, proteins, is still important this new data sources have led to a shift of focus to the study of biological systems in their entirety. In doing so we realised that the most important aspect of these systems is not the inner workings of the individual components but rather the interactions between them.

Traditionally biology has not used any formal methods to explain things and perhaps that was not needed when we were dealing with single entities. As the systems grew in size it became apparent that some change in direction was needed to handle the increased complexity and get a coherent story. Systems Biology tries to fill this gap by bringing in practises from other more traditionally formal fields like Mathematics, Physics, Computer Science(Ideker et al. [1], Kitano [3]).

Early attempts to capture the more complex picture painted by the increase in the number of components were to use static diagrams to capture the interactions and dependencies between them. These diagrams have been widely used in Biochemistry and particularly in metabolism since the species conversion process that takes place in metabolic pathways can easily be seen as a path or paths through these diagrams going from one metabolite to the next(see for example Figure 1.1). Now these diagrams are very important and they contain important knowledge about our understanding of the operations of biological systems. There exist databases that contain our current understanding of a particular system annotated with information from different sources and these are particularly popular for metabolic pathways (see Kanehisa and Goto [2], Pharkya et al. [4]). Despite the importance of such diagrams they only give partial information as they paint a very static picture of the system.

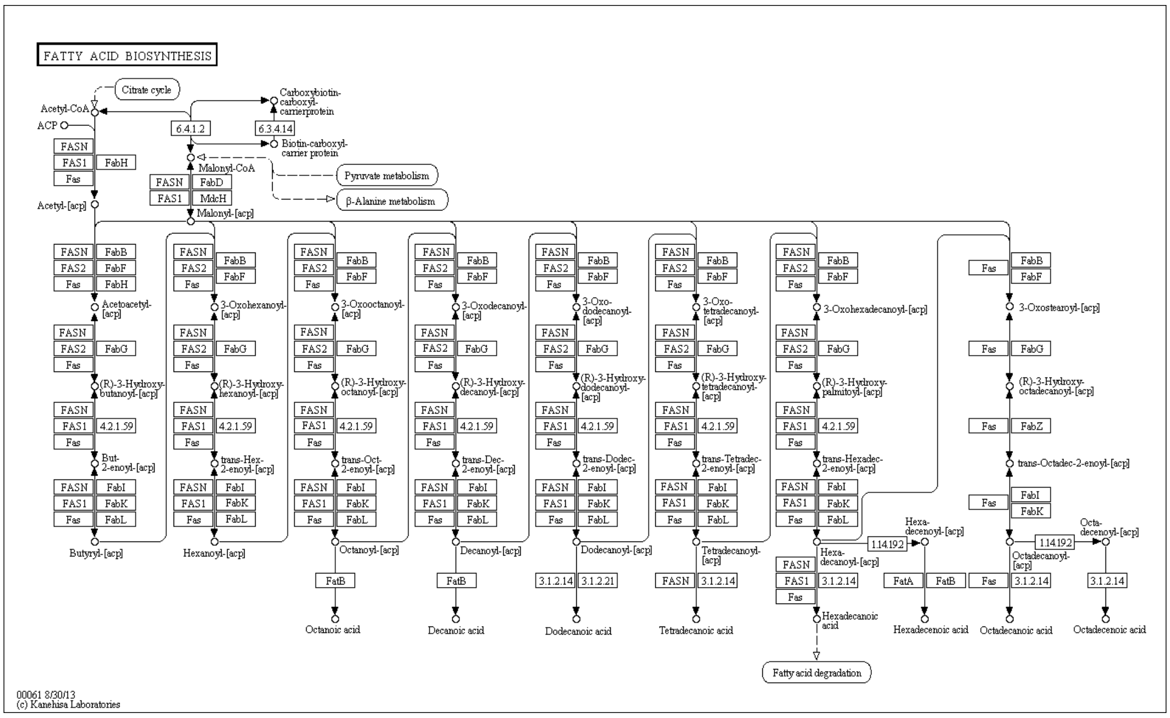

Fig. 1.1 Fatty acid synthesis pathway captured by the standard diagrammatic language used to capture interactions between components in a biological system. This was taken from KEGG a database which contains current knowledge about the working of these systems(from experiments for example) integrated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

1.1 Dynamical systems theory and Flux Balance Analysis

1.2 Challenge of lipid metabolism

1.3 Computational models in Biology

1.4 Outline of work

Chapter 2

Methods

2.1 Petri Nets

2.1.1 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* : $T \to \mathbf{m}P$ which assigns a multiset of places $pre(t)$ to each transition $t \in T$
- a *postconditions map post* : $T \to \mathbf{m}P$ 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 $f : P \to \mathbb{N}$. 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 $p \rightarrow t$ if $pre(t)_p = n$. A weighted directed edge with weight $n > 0$ is added between transition t and place p in the direction $t \rightarrow p$ 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 $post(t)_p > 0$.

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

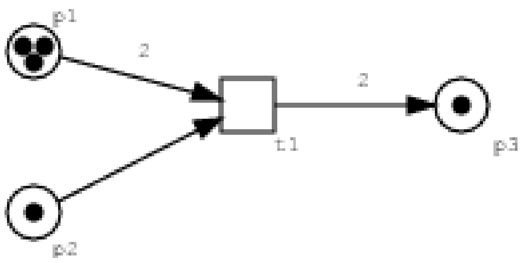

Fig. 2.1 Small basic Petri Net example

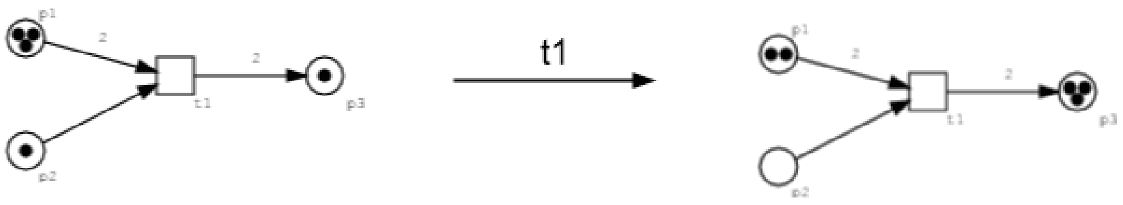

Fig. 2.2 The token game for basic Petri Nets.

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.

2.1.2 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 $(P, T, pre, post, v)$ with everything defined as before with the addition of the map $v : T \to H$ 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.

2.1.3 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:

$$2A + 3B \xrightarrow{r_k} 5C$$

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:

2.2 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] Ideker, T., Galitski, T., and Hood, L. (2001). A new approach to decoding life: systems biology. *Annual review of genomics and human genetics*, 2(1):343–372.

[2] Kanehisa, M. and Goto, S. (2000). Kegg: kyoto encyclopedia of genes and genomes. *Nucleic acids research*, 28(1):27–30.

[3] Kitano, H. (2002). Systems biology: a brief overview. *Science*, 295(5560):1662–1664.

[4] Pharkya, P., Nikolaev, E. V., and Maranas, C. D. (2003). Review of the brenda database. *Metabolic engineering*, 5(2):71–73.

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.