Proceedings of the Strasbourg Spring School
on

advances in Systems and Synthetic Biology

March 23rd - 27th, 2015

Edited by

Patrick Amar, François Képès, Vic Norris

"But technology will ultimately and usefully be better served by following the spirit of Eddington, by attempting to provide enough time and intellectual space for those who want to invest themselves in exploration of levels beyond the genome independently of any quick promises for still quicker solutions to extremely complex problems."

Strohman RC (1977) Nature Biotech 15:199

FOREWORD

Systems Biology includes the study of interaction networks and, in particular, their dynamic and spatiotemporal aspects. It typically requires the import of concepts from across the disciplines and crosstalk between theory, benchwork, modelling and simulation. The quintessence of Systems Biology is the discovery of the design principles of Life. The logical next step is to apply these principles to synthesize biological systems. This engineering of biology is the ultimate goal of Synthetic Biology: the rational conception and construction of complex systems based on, or inspired by, biology, and endowed with functions that may be absent in Nature.

This annual School started in 2002. It was the first School dedicated to Systems Biology in France, and perhaps in Europe. Since 2005, Synthetic Biology has played an increasingly important role in the School. Generally, the topics covered by the School have changed from year to year to accompany and sometimes precede a rapidly evolving scientific landscape. Its title has evolved in 2004 and again in 2012 to reflect these changes. The first School was held near Grenoble after which the School has been held in various locations. It started under the auspices of Genopole®, and has been supported by the CNRS since 2003, as well as by several other sponsors over the years.

This book gathers overviews of the talks, original articles contributed by speakers and students, tutorial material, and poster abstracts. We thank the sponsors of this conference for making it possible for all the participants to share their enthusiasm and ideas in such a constructive way.

Patrick Amar, Gilles Bernot, Marie Beurton-Aimar, Attila Csikasz-Nagy, Jürgen Jost, Ivan Junier, Marcelline Kaufman, François Képès, Pascale Le Gall, Sheref Mansy, Jean-Pierre Mazat, Victor Norris, William Saurin, El Houssine Snoussi, Ines Thiele, Birgit Wiltschi.

ACKNOWLEDGEMENTS

We would like to thank the conference participants, who have contributed in a way or another to this book. It gathers overviews of the talks, discussions, original articles and tutorial material contributed by speakers, abstracts from attendees, short articles from students, posters and lectures proposed by the epigenesis groups to review or illustrate matters related to the scientific topic of the conference.

Of course the organisation team would like to express gratitude to all the staff of the *Holiday Inn* hotel and to the *École Supérieure de Biotechnologie Strasbourg* for the very good conditions we have found during the conference.

Special thanks to the Epigenomics project for their assistance in preparing this book for publication. The cover photography shows a view of the *Petite France*, at Strasbourg.

We would also like to express our thanks to the sponsors of this conference for their financial support allowing the participants to share their enthusiasm and ideas in such a constructive way.

They were:

- Centre National de la Recherche Scientifique (CNRS):
  http://www.cnrs.fr
- Genopole® Évry:
  http://www.genopole.fr
- GDRE CNRS 513 Biologie Systémique:
  http://www.mpi-magdeburg.mpg.de/CNRS_MPG
- Université d'Évry Val d'Essonne:
  http://www.univ-evry.fr/
- Institut National de Recherche en Informatique et en Automatique (INRIA):
  http://www.inria.fr/
- Université de Strasbourg:
  http://www.unistra.fr/
- École Supérieure de Biotechnologie Strasbourg (ESBS):
  http://www.esbs.u-strasbg.fr/
- GDR CNRS 3003 Bioinformatique Moléculaire:
  http://www.gdr-bim.u-psud.fr
- Laboratoire des Sciences de l'Ingénieur, de l'Informatique et de l'Imagerie (ICube):
  http://icube.unistra.fr
- Télécom Physique Strasbourg:
  http://www.telecom-physique.fr

THE EDITORS

INVITED SPEAKERS

Laurence CALZONE	Inst. Curie, Paris, F
Joel BADER	Johns Hopkins U., Baltimore, US
Damien COUDREUSE	IGDR, U. Rennes, F
Jacques DEMONGEOT	AGIM, Grenoble, F
Oliver EBENHOEH	ICSMB, U. Aberdeen, UK
François FAGES	INRIA, Rocquencourt, F
Albert GOLDBETER	ULB, Bruxelles, BE
Jens HAUSER	Synth-ethic, Vienna, AT
Nicolas LENOVÈRE	Babraham Institute, Cambridge, UK
Anne OSBOURN	John Innes Centre, Norwich, UK
Magali REMAUD	LISBP / Toulouse White Biotech, F
Jürgen ZANGHELLINI	ACIB, Graz, AT

CONTENTS

PART I INVITED TALKS 11

CELL MODELLING

MODELLING BIOLOGICAL NETWORKS

PLANT SYSTEMS & SYNTHETIC BIOLOGY

YEAST SYSTEMS & SYNTHETIC BIOLOGY

ENZYME & METABOLIC MODELLING

PART I INVITED TALKS

System Biology of Cellular Rhythms: Modeling the Dynamics of the Mammalian Cell Cycle

Albert GOLDBETER[1]

[1] ULB, B-1050 Brussels, Belgium

Abstract

Cellular rhythms originate from the regulatory feedback loops that control the dynamics of biochemical processes and represent a phenomenon of temporal self-organization. They illustrate how an emergent property, autonomous oscillatory behavior, arises from molecular interactions in regulatory networks. This explains why oscillatory phenomena abound at the cellular level. After providing an overview of biological rhythms and of their underlying mechanisms, I will focus on the cell cycle, which provides a major example of rhythmic behavior at the cellular level.

The mammalian cell cycle, driven by an enzymatic network of cyclin-dependent kinases, behaves as a self-sustained oscillator. A detailed computational model shows that the regulatory structure of this network results in its temporal self-organization in the form of sustained oscillations that bring about the orderly progression along cell cycle phases. The coupling of the cell cycle to the circadian clock results in the synchronization of these two major cellular rhythms. To understand the dynamics of the cell cycle we need to characterize the balance between cell cycle arrest and cell proliferation, which is often deregulated in cancers. We address this issue by means of the detailed computational model for the network of cyclin-dependent kinases (Cdks) driving the mammalian cell cycle.

Previous analysis of the model focused on how this balance is controlled by growth factors or by the levels of activators (oncogenes) and inhibitors (tumor suppressors) of cell cycle progression. Suprathreshold changes in the level of any of these factors can trigger a switch in the dynamical behavior of the Cdk network corresponding to a bifurcation between a stable steady state, associated with cell cycle arrest, and sustained oscillations of the various cyclin/Cdk complexes, corresponding to cell proliferation. Cell proliferation can also be controlled by cellular environmental factors external to the Cdk network, such as the extracellular matrix, and contact inhibition, which increases with cell density. Whether the balance in the Cdk network is tilted toward cell cycle arrest or proliferation depends on the direction in which the threshold associated with the bifurcation is passed once the cell integrates the multiple, internal or external signals that promote or impede progression in the cell cycle.

References

[1] Goldbeter A (1996) *Biochemical Oscillations and Cellular Rhythms. The molecular bases of periodic and chaotic behaviour.* Cambridge Univ. Press, Cambridge, UK.

[2] Goldbeter A (2002) Computational approaches to cellular rhythms. *Nature* **420**, 238-245.

[3] Gérard C & Goldbeter A (2009) Temporal self-organization of the cyclin/Cdk network driving the mammalian cell cycle. *Proc Natl Acad Sci USA* **106**, 21643-21648.

[4] Goldbeter A (2010) *La Vie oscillatoire. Au coeur des rythmes du vivant* (Odile Jacob, Paris).

[5] Gérard C & Goldbeter A (2012) Entrainment of the mammalian cell cycle by the circadian clock : Modeling two coupled cellular rhythms. *PLoS Comput. Biol.* **8 (5)**: e1002516.

[6] Gérard C & Goldbeter A (2014) The balance between cell cycle arrest and cell proliferation: control by the extracellular matrix and by contact inhibition. *Interface Focus* **4**: 20130075.

From a biological question to a mathematical model: an example of understanding patterns of genetic alterations in bladder tumorigenesis

Laurence CALZONE[1]

[1] Institut Curie, Paris, F

Abstract

I will start introducing our systems biology approach, from the definition of the question to the construction of a mathematical model and the formulation of predictions. I will then show how we applied our approach to bladder cancer and attempted to explain the co-occurrence and mutual exclusivity of genetic alterations in a set of genes frequently mutated in bladder tumours.

Bladder tumours progress along two main pathways: the CIS pathway and the Ta pathway. The Ta pathway, less aggressive than the CIS pathway, is characterized by a high frequency of activating fibroblast growth factor receptor 3 (FGFR3) gene mutations, which are rare in the CIS pathway. We combined mathematical modelling and statistical analyses to better understand the diverse alterations observed in bladder tumorigenesis. In a dataset of 178 patients including both invasive and non-invasive tumours, we performed statistical tests on a list of genes known to be frequently altered in bladder cancer in order to identify co-occurrences or exclusivities between all these alterations. We focused on genetic alterations (mutations, homozygous losses, and amplifications) of genes frequently mutated in bladder cancer. We identified 9 associations and verified them in 3 other public datasets. We then constructed a logical model of cell cycle and apoptosis entry in order to explain the context for these patterns of genetic alterations. With the model, we formulated some predictions that we verified back in each of the three datasets when possible. Finally, we explored a method linking our transcriptomics dataset of the 178 tumours to the stable states of our logical model and confirmed that the invasive tumours are associated with proliferative stable states and non-invasive tumours with apoptosis. We attempted to stratify patients based on the solutions of the mathematical model.

Genetic regulatory networks: focus on attractors of their dynamics

Jacques DEMONGEOT[1]

[1] UJF & IUF, Grenoble, F

Abstract

Genetic regulatory networks are devoted to the control and maintenance of important functions like the the energy control of cells, and the morphogenesis and defence of living organisms. Since the innate system of defence represented by the Toll Like Receptors (TLR, already present in insects), mammals have developed an adaptive immune system during the embryonic maturation of their T Cells Receptors α and β ($TCR\alpha$ and $TCR\beta$) from strategies of DNA rearrangements essentially under the control of the RAG gene. We will describe the immunologic networks (called "immunetworks") in charge of controlling the concentration of both TLR's and TCR's and show that the circuits in the core of their interaction graphs are responsible of a few number of dedicated attractors, responsible of the dynamics of receptors synthesis.

In the same spirit, we will describe a genetic network important for the oxidative metabolism of the cell, the Ferritin (the iron-storage protein) control network regulating the iron metabolism in mammals and eventually study the Engrailed morphogenetic network.

From art to engineering: 15 years of standards and tools towards synthetic and digital organisms

Nicolas LENOVÈRE[1]

[1] Babraham Institute, Cambridge, UK

Abstract

Cells as Machines, Reactions as Programs: a Computer Science Approach towards Mastering the Complexity of Cell Processes[1]

François Fages*

* Inria Paris-Rocquencourt, Lifeware team, France
http://lifeware.inria.fr/

Abstract

Systems biology aims at understanding complex biological processes in terms of their elementary mechanisms at the molecular level. The bet of applying computer science concepts and tools for the analysis of biochemical reaction systems in the cell, designed by natural evolution, has led to novel model-based insights in cell biology and new challenges in computer science. In this course, we shall review the development over the last decade of the biochemical abstract machine (BIOCHAM) software environment for modeling molecular reaction systems, reasoning about them at different levels of abstraction, formalizing biological behaviors in temporal logic, inferring kinetic parameter values, measuring robustness, and start deciphering natural biochemical programs in the cell.

1 Introduction

At the end of the 90s, with the end of the human genome project, research in bioinformatics started to evolve, passing from the analysis of the genomic sequence and structural biology problems, to the analysis of complex post-genomic interaction networks: expression of RNA and proteins, protein-protein interactions, transport, signal transduction, cell cycle, etc. Systems biology is the name given to a new pluridisciplinary research field, involving biologists, computer scientists, mathematicians, physicists, to promote a change of focus towards system-level understanding of high-level functions of living organisms, from their biochemical bases at the molecular level. The main outcome of this effort has been the creation of, and easy access to, databases and ontologies of cell components; repositories of models of cell processes such as BioModels.net, through the definition of common exchange formats such as the Systems Biology Markup Language (SBML); model editors and simulation tools, making it possible to reproduce *in silico* analyses in articles, with models published as supplementary material; and the construction of

[1] These lecture notes are extracted from [1].

a whole cell predictive computational model of the bacterium *Mycoplasma genitalium* including its 525 genes by Karr et al. in 2012.

In this domain, formal methods from computer science have been successfully applied to master the complexity of biological networks and decipher biological processes, mostly at the molecular and cellular levels. The distinction between syntax and semantics is particularly fruitful for designing modeling languages and for reasoning about biological systems at different levels of abstraction. While interaction diagrams are the key for interacting with biologists, their transcription in formal graphs or formal languages compels the modeler to eliminate any ambiguity, and enables the use of a wide variety of structural or dynamic analysis tools. In these approaches, the mathematical formalisms of ordinary differential equations (ODE) and partial derivate equations (PDE) appear as low-level languages on top of which high-level languages can be designed to directly reflect the structure of the interactions, and apply novel static analysis methods. The notion of Petri net transition-invariant is a key tool for analyzing extreme fluxes and optimizing metabolic networks [2]. Place-invariants provide structural conservation laws that can be directly used to eliminate variables in mathematical models based on ordinary differential equation models. The notion of siphons and traps provide sufficient conditions for persistence and accumulation of molecular species in a network of reactions [3, 4].

In this paper, we review the development over the last decade of the Biochemical Abstract Machine (BIOCHAM[2]) software environment for modeling cell biology molecular reaction systems, reasoning about them at different levels of abstraction, formalizing biological behaviors in temporal logic with numerical constraints, and using them to infer non-measurable kinetic parameter values, evaluate robustness, decipher natural biochemical processes and design new biochemical programs in synthetic biology.

2 Biochemical Reaction Systems

Let $\mathcal{S}$ be a finite set of s molecular species. A reaction is a triple $(\boldsymbol{s}, \boldsymbol{s'}, f)$, noted $\boldsymbol{s} \xrightarrow{f} \boldsymbol{s'}$, where $\boldsymbol{s}, \boldsymbol{s'} : \mathcal{S} \to \mathbb{N}$ are multisets over $\mathcal{S}$ (stoichiometric coefficients), and $f : \mathbb{R}^s \to \mathbb{R}$ is a mathematical function over molecule quantities, called the rate function. Multisets are used for representing reactants and products in reactions, and a reaction is fundamentally a multiset rewriting rule. The chemical metaphor based on multiset rewriting has been proposed in computer science to program concurrent processes and reason about them [5]. However in biochemistry, time matters, the reaction rates of the reactions may

[2] http://contraintes.inria.fr/biocham

differ by several orders of magnitude, and it is crucial for many properties to consider the continuous-time dynamics of the reactions. Each reaction is thus supposed to be given with a rate function.

A limited number of reaction schemas occurs in biochemical reaction networks. *Binding* reactions of the form

$$A, B \xrightarrow{kAB} C$$

bind two molecular compounds together, such as the *complexation* of two proteins or complexes to form a bigger complex, or the binding of a promotion factor (resp. an inhibitor) on a gene to activate (resp. inhibit) its transcription. The mass action law kinetics used in that reaction states that the rate of the reaction is proportional to the number of its reactants. The rate constant k represents the affinity of the two molecules to bind together. The inverse unbinding reaction is of the form

$$C \xrightarrow{k.C} A, B$$

with again a mass action law kinetics, where the rate constant characterizes the stability of the complex.

A molecular species like a protein can also be modified under the action of an enzyme, such as a kinase for a *phosphorylation* reaction, or a phosphatase for a dephosphorylation reaction. This is represented by a reaction of the form

$$A \xrightarrow{v.A/(k+A)} B$$

with a Michaelis-Menten kinetics, which comes in fact for the reduction of three elementary reactions with mass action law kinetics
$(A, E \underset{k_2 C}{\overset{k_1.A.E}{\rightleftarrows}} C \xrightarrow{k_3.C} B, E)$ by quasi-steady state approximation.
Synthesis reaction, such as the synthesis of an RNA by a gene activated by its promotion factor, are of the form

$$A \xrightarrow{v.A^n/(k+A)^n} A, B$$

with a Hill kinetics of order n. That rate function provides a sigmoidal response, i.e. a switch-like behavior to the synthesis process, and comes from the reduction of a system of n cooperative reactions.

Degradation reactions of the form

$$A \xrightarrow{k.A} \_$$

have the empty multiset as product, and either a mass action law kinetics in the case of spontaneous degradation, or a Michaelis-Menten or Hill kinetics in the case of an active degradation process under the action of other molecules.

These formal systems of reactions can be interpreted at different level of abstraction in a hierarchy of semantics. The most concrete interpretation is provided by the *Chemical Master Equation* (CME), which defines the probability of being in a state $\boldsymbol{x}$ at time t as

$$\frac{d}{dt}p^{(t)}(\boldsymbol{x}) = \sum_{j:\boldsymbol{x}-\boldsymbol{r}_j \geq 0} f_j(\boldsymbol{x}-\boldsymbol{v}_j).p^{(t)}(\boldsymbol{x}-\boldsymbol{v}_j) - \sum_{j=1}^{n} f_j(\boldsymbol{x}-\boldsymbol{v}_j).p^{(t)}(x)$$

where $\boldsymbol{v}_j$ is the change vector $\boldsymbol{s'}_j - \boldsymbol{s}_j$ of reaction j and $f_j(\boldsymbol{x})$ is the propensity of reaction j in state $\boldsymbol{x}$ defined by the rate function.

The *continuous semantics* of a reaction system is a deterministic interpretation, which describes the time evolution of the mean $E[X(t)]$ by an ODE. The ODE derives from the CME by a first-order approximation. We have

$$\frac{d}{dt}E[X(t)] = \sum_{\boldsymbol{x}} \frac{d}{dt}p^{(t)}(\boldsymbol{x}) = \sum_{j=1}^{n} \boldsymbol{v}_j.E[f(X(t))]$$

which gives, by first-order approximation of the Taylor series about the mean,

$$\frac{d}{dt}\boldsymbol{\mu} = \sum_{j=1}^{n} \boldsymbol{v}_j.f(\boldsymbol{\mu}).$$

Given initial concentrations for species, such an ODE can be simulated by standard numerical methods for stiff systems.

The *stochastic semantics* of a reaction system is defined by a Continuous Time Markov Chain (CTMC) over integer numbers of molecules (discrete concentration levels). The rate functions of the reactions lead to state transition probabilities after normalization by the sum of the propensities of each reaction in each state. The Stochastic Simulation Algorithm of Gillespie provides a simulation method which computes numerical traces, most often similar to the ODE simulation for large numbers of molecules, but may exhibit qualitatively different behaviors in the case of small numbers of molecules, for instance in the case of gene expression as a gene usually is in one single copy in a cell.

The abstraction of the stochastic semantics by simply forgetting the probabilities, gives the non-deterministic *Petri net semantics* of the reactions, where the discrete states define the number of tokens in each place, and the transitions consume the reactant tokens and produce the product tokens. The abstraction of the Petri net semantics in the *Boolean semantics* defined by the Boolean abstraction function over integers, $\beta : \mathbb{N} \longrightarrow \{0,1\}$ with $\beta(0) = 0$ and $\beta(x) = 1$ if $x > 0$, is a non-deterministic asynchronous Boolean transition system suitable for reasoning on the presence/absence of molecules.

In BIOCHAM, the Boolean semantics of the reactions associates several Boolean transitions to one reaction. For instance, a complexation reaction like $A, B \longrightarrow B$, is interpreted by 4 Boolean transitions, one for each possible complete consumption of the 2 reactants: $A \wedge B \longrightarrow C \wedge \pm A \wedge \pm B$. This is necessary for the abstraction result to hold with respect to the Petri net or stochastic semantics.

In [6], all these discrete and stochastic trace semantics of reactions systems have been related by formal abstraction relationships (Galois connections) in the framework of abstract interpretation. This shows that if a behavior is not possible in the Boolean semantics for instance, then it is not realizable in the Petri net or stochastic semantics for any kinetic laws and kinetic parameter values.

3 Symbolic Model-Checking of Biochemical Systems

A Boolean state specifies the presence or absence of each molecule in the system at a given time, and any set of states can be represented by a Boolean constraint over the molecule variables. The *Computation Tree Logic* CTL* is a modal logic that extends propositional logic with two path quantifiers, $\mathbf{A}$ and $\mathbf{E}$ ($\mathbf{A}\phi$ meaning that ϕ is true on all computation paths, and $\mathbf{E}\phi$ that it is true on at least one path), and several temporal operators, $\mathbf{X}\phi$ (meaning that ϕ is true on the next state on a path), $\mathbf{F}\phi$ (meaning that ϕ is finally true on some state on a path), $\mathbf{G}\phi$ (globally true on all states on a path), $\phi\mathbf{U}\psi$ (until, meaning that ψ is finally true and ϕ is always true before), and $\phi\mathbf{R}\psi$ (release, meaning that ψ is either globally true or always true up to the first occurrence of ψ included). In this logic, $F\phi$ is equivalent to $true\mathbf{U}\phi$, $G\phi$ to $\phi\mathbf{R}false$, and we have the following duality properties: $\neg\mathbf{X}\phi = \mathbf{X}\neg\phi$, $\neg\mathbf{E}\phi = \mathbf{A}\neg\phi$, $\neg\mathbf{F}\phi = \mathbf{G}\neg\phi$, $\neg(\phi\mathbf{U}\psi) = \neg\psi\mathbf{R}\neg\phi$.

The fragment CTL of CTL* imposes that a temporal opertor must immediately follow a path quantifier. This logic CTL can express a wide variety of properties of biochemical networks like state *reachability* of ϕ ($\mathbf{EF}\phi$), *steadyness* of ϕ ($\mathbf{EG}\phi$), *stability* ($\mathbf{AG}\phi$), reachability of a stable state ($\mathbf{EFAG}\phi$), ϕ *checkpoint* for ψ ($\neg\psi\mathbf{R}\phi$), *oscillations* ($\mathbf{EG}(\mathbf{F}\neg\phi \wedge \mathbf{F}\phi)$ over-approximated in CTL by $\mathbf{EG}(\mathbf{EF}\neg\phi \wedge \mathbf{EF}\phi)$) etc.

Our first result in [7] was to show the performance of a state-of-the-art symbolic model checker using the representation of Boolean formulae by ordered binary decision diagrams (OBDD), on Kohn's map of the mammalian cell cycle. This map was transcribed in a reaction model of 732 reaction rules over 165 proteins and genes, and 532 variables taking into account the different forms of the molecular species. The astronomical number of Boolean states in this system, 2^{532}, prevents the explicit representation of the state graph, however, a set of states in this space can nevertheless be represented symbolically by a Boolean formula over 532 variables, and the transition relation by a Boolean formula over twice that number of variables. For instance the formula $false$ represents the empty set, $true$ the universe of all states, x the set of 2^{531} states where x is present, etc. The compilation of the whole 732 reactions into Boolean formulae took 29 seconds, and simple reachability and oscillations properties could be checked in a few seconds. The negative answer to the query concerning the oscillation of cyclin B revealed an omission of the synthesis of cyclin B in Kohn's map.

The encoding of biological properties in temporal logics provides a *logical paradigm for systems biology* that makes a bridge between theoretical models and biological experiments, through the following identifications:

biological model = transition system,
biological property = temporal logic formula,
model validation = model-checking,
model inference = constraint solving.

A formula ϕ, learned from biological experiments, can be tested in a model $\mathcal{M}$ by model-checking techniques to determine whether $\mathcal{M} \models \phi$. Furthermore, a model-checker can also compute the set of initial states for which a formula is true, and suggest biological experiments to verify a CTL property predicted by the model [8].

4 Quantitative Temporal Logic Constraints

4.1 Threshold and Timing Constraints

The temporal logic approach to the specification of imprecise dynamical properties of biological systems can also be made quantitative and applied to quantitative models over concentrations. The idea is to lift it to a first-order setting with numerical (linear) constraints over the reals, in order to express threshold or more complex constraints on the concentrations of the molecular compounds and time [9].

For instance, the reachability of a threshold concentration for a molecule A can be expressed with the formula $\mathbf{F}(A > v)$ for some value or free variable v. Such formulae can then be interpreted on a finite numerical trace (extended with a loop on the last state) obtained either from a biological experiment, or from the numerical simulation of an ODE model.

In BIOCHAM, we use the First-Order Linear Time Logic with linear constraints over the reals (FO-LTL($\mathbb{R}_{\mathrm{lin}}$)) to specify semi-qualitative semi-quantitative properties of a biological dynamical system. LTL is the fragment of CTL* without any path quantifier and only time operators interpreted on a trace. The grammar of FO-LTL($\mathbb{R}_{\mathrm{lin}}$) formulae is

$$\phi ::= \ \ \mathrm{c} \ | \ \phi \Rightarrow \psi \ | \ \phi \wedge \phi \ | \ \phi \vee \phi \ | \ \mathbf{X}\phi \ | \ \mathbf{F}\phi \ | \ \mathbf{G}\phi \ | \ \phi\mathbf{U}\phi \ \ | \ \phi\mathbf{R}\phi$$

Timing constraints can be expressed with the time variable and free variables to relate the time of differents events. For instance, the formula $\mathbf{G}(Time \leq t_1 \Rightarrow [A] < 1 \wedge Time \geq t_2 \Rightarrow [A] > 10) \wedge (t_2 - t_1 < 60)$ expresses that the concentration of molecule A is always less than 1 up to some time t_1, always greater than 10 after time t_2, and the switching time between t_1 and t_2 is less than 60 units of time. A local maximum for molecule concentration A can be defined with the formula $\mathbf{F}(A \leq x \wedge \mathbf{X}(A = x \wedge \mathbf{X}A \leq x))$. This formula can be used to define oscillation properties, with period constraints defined as time separation constraints between the local maxima of the molecule, as well as phase constraints between different molecules.

In [10], it is shown how the *validity domain* $\mathcal{D}_{(s_0,\ldots,s_n),\phi}$ of the free variables of an FO-LTL($\mathbb{R}_{\mathrm{lin}}$) formula ϕ on a finite trace $(s_0, ..., s_n)$, can be computed by finite unions and intersections of polyhedra.

4.2 Parameter Optimization and Robustness

One major difficulty in quantitative systems biology, is that the kinetic parameter values of the biochemical reactions are usually unknown, and must be infered from the observable behavior of the system under various conditions (differences of milieu, drugs, gene knock-outs or knock downs, etc.).

In our quantitative temporal logic setting, this problem amounts to solve the inverse problem of finding parameter values for an ODE model such that an FO-LTL($\mathbb{R}_{\mathrm{lin}}$) specification is true. However, the classical true/false valuation of a logical formula is not well suited to guide the search. State-of-the-art continuous optimization algorithms such as evolutionary algorithms, require a fitness function to measure progress towards satisfiability. Such a continuous satisfaction degree in the interval $[0, 1]$ can be defined for FO-LTL($\mathbb{R}_{\mathrm{lin}}$) formulae, by replacing constants by variables, which was in fact our original motivation for considering formulae with free variables.

Indeed, a specification of the expected behavior given by a closed formula, for instance $\phi_2 = \mathbf{F}(A \geq 7 \wedge \mathbf{F}(A \leq 0))$, can first be abstracted in a formula with free variables by replacing constants with free variables, e.g. $\phi = \mathbf{F}(A \geq y_1 \wedge \mathbf{F}(A \leq y_2))$ with the objective values 7 for y_1 and 0 for y_2. Then, the validity domain $\mathcal{D}_{T,\phi}$ of the formula ϕ on a trace T obtained by simulation for some parameter values, makes it possible to define the *violation degree* $vd(T, \phi, o)$ of the formula on T with objective o, simply as the distance between the validity domain and the objective point o. A *continuous satisfaction degree* in the interval $[0, 1]$ can then be defined by normalization as the inverse of the violation degree d plus one,

$$sd(T, \phi, o) = \frac{1}{1 + vd(T, \phi, o)}$$

In BIOCHAM, we use the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) of N. Hansen [11] as a black-box optimization algorithm, with the satisfaction degree of an FO-LTL($\mathbb{R}_{\mathrm{lin}}$) specification as fitness function, and unknown kinetic parameter values (initial concentrations and control parameters) as variables. This strategy for optimizing parameters with respect to an FO-LTL($\mathbb{R}_{\mathrm{lin}}$) specification allowed us to solve a wide variety of problems in systems biology, for fitting models to experimental data in high dimension (up to 100 parameters), revisiting the structure of the reaction network in case of failure, making new biological hypotheses based on simulation, and verifying them by new experiments, for instance for deciphering the complex dynamics of a cell signaling network in [12]. The same strategy for parameter optimization can also be used to compute control parameters to achieve a desired behavior at the single cell of cell population levels. This has been used for the model-based real-time control of gene expression in yeast cells using a microfluidic device in [13], and at the whole body scale, to couple models of cell cycle, circadian clock, drug effects, DNA repair system, and optimize anti-cancer drug chronotherapeutics in [14].

Kitano gives a general definition of the robustness of a property ϕ of a system S with respect to a set P of perturbations given with their probability distribution, as the mean functionality of the system with respect to ϕ under the perturbations, with the system's functionality defined in an *ad hoc* way for each property. In our framework, this definition can be instanciated to a complete definition for FO-LTL($\mathbb{R}_{\mathrm{lin}}$) properties, simply by taking their continuous satisfaction degree as functionality measure, as follows [15]:

$$\mathcal{R}_{S,\phi,P} = \int_{p \in P} prob(p)\ sd(T_p, \phi)\ dp.$$

This definition of robustness can then be evaluated in a model by 1) sampling the perturbations according to their distribution; 2) measuring the satisfaction degree of the property for each simulation of the perturbed model; and 3) returning the average satisfaction degree.

5 Conclusion

This line of research in systems biology based on the vision of cell as computation, aims at mastering the complexity of cell processes, through the use of concepts and tools from computer science and the establishment of formal computation paradigms tightly coupled to experimental settings in cell biology. While for the biologist, as well as for the mathematician, the sizes of the biological networks and the number of elementary interactions constitute a complexity barrier, for the computer scientist the difficulty is not that much in the size of the networks than in the unconventional nature of biochemical computation. Unlike most programs, biochemical computation involve transitions that are stochastic rather than deterministic, continuous-time rather than discrete-time, poorly localized in compartments instead of well-structured in modules, and created by evolution instead of by rational design. It is our belief however that some form of modularity (functional if not structural) is required by an evolutionary system to survive, and that the elucidation of these modules in biochemical computation is now a key to master the analog aspects of biochemical computation, understand natural biochemical programs, and start controlling the cell machinery.

References

[1] F. Fages, "Cells as machines: towards deciphering biochemical programs in the cell (invited talk)," in *Proc. 10th International Conference on Distributed Computing and Internet Technology ICDCIT'14* (R. Natarajan, ed.), vol. 8337 of *Lecture Notes in Computer Science*, pp. 50–67, Springer-Verlag, 2014.

[2] I. Zevedei-Oancea and S. Schuster, "Topological analysis of metabolic networks based on petri net theory," *In Silico Biology*, vol. 3, no. 29, 2003.

[3] D. Angeli, P. D. Leenheer, and E. D. Sontag, "A petri net approach to persistence analysis in chemical reaction networks," in *Biology and Control Theory: Current Challenges*, vol. 357 of *LNCIS*, pp. 181–216, Springer-Verlag, 2007.

[4] F. Nabli, F. Fages, T. Martinez, and S. Soliman, "A boolean model for enumerating minimal siphons and traps in petri-nets," in *Proceedings of CP'2012, 18th International Conference on Principles and Practice of Constraint Programming*, vol. 7514 of *Lecture Notes in Computer Science*, pp. 798–814, Springer-Verlag, Oct. 2012.

[5] G. Berry and G. Boudol, "The chemical abstract machine," *Theoretical Computer Science*, vol. 96, 1992.

[6] F. Fages and S. Soliman, "Abstract interpretation and types for systems biology," *Theoretical Computer Science*, vol. 403, no. 1, pp. 52–70, 2008.

[7] N. Chabrier-Rivier, M. Chiaverini, V. Danos, F. Fages, and V. Schächter, "Modeling and querying biochemical interaction networks," *Theoretical Computer Science*, vol. 325, pp. 25–44, Sept. 2004.

[8] G. Bernot, J.-P. Comet, A. Richard, and J. Guespin, "A fruitful application of formal methods to biological regulatory networks: Extending Thomas' asynchronous logical approach with temporal logic," *Journal of Theoretical Biology*, vol. 229, no. 3, pp. 339–347, 2004.

[9] F. Fages and P. Traynard, "Temporal logic modeling of dynamical behaviors: First-order patterns and solvers," in *Logical Modeling of Biological Systems* (L. F. del Cerro and K. Inoue, eds.), ch. 8, pp. 291–323, John Wiley & Sons, Inc., 2014.

[10] A. Rizk, G. Batt, F. Fages, and S. Soliman, "Continuous valuations of temporal logic specifications with applications to parameter optimization and robustness measures," *Theoretical Computer Science*, vol. 412, no. 26, pp. 2827–2839, 2011.

[11] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary Computation*, vol. 9, no. 2, pp. 159–195, 2001.

[12] D. Heitzler, G. Durand, N. Gallay, A. Rizk, S. Ahn, J. Kim, J. D. Violin, L. Dupuy, C. Gauthier, V. Piketty, P. Crépieux, A. Poupon, F. Clément, F. Fages, R. J. Lefkowitz, and E. Reiter, "Competing G protein-coupled receptor kinases balance G protein and β-arrestin signaling," *Molecular Systems Biology*, vol. 8, June 2012.

[13] J. Uhlendorf, A. Miermont, T. Delaveau, G. Charvin, F. Fages, S. Bottani, G. Batt, and P. Hersen, "Long-term model predictive control of gene expression at the population and single-cell levels," *Proceedings of the National Academy of Sciences USA*, vol. 109, no. 35, pp. 14271–14276, 2012.

[14] E. De Maria, F. Fages, A. Rizk, and S. Soliman, "Design, optimization, and predictions of a coupled model of the cell cycle, circadian clock, dna repair system, irinotecan metabolism and exposure control under temporal logic constraints," *Theoretical Computer Science*, vol. 412, pp. 2108–2127, May 2011.

[15] A. Rizk, G. Batt, F. Fages, and S. Soliman, "A general computational method for robustness analysis with applications to synthetic gene networks," *Bioinformatics*, vol. 12, pp. il69–il78, June 2009.

Unlocking plant metabolic diversity

Anne OSBOURN[1]

[1] John Innes Centre, Norwich, UK

Abstract

Plants produce a tremendous array of natural products, including medicines, flavours, fragrances, pigments and insecticides. The vast majority of this metabolic diversity is as yet untapped, despite its huge potential value for humankind. The recent discovery that genes for the synthesis of different kinds of natural products are organised in clusters in plant genomes is now opening up opportunities for systematic mining for new pathways and chemistries. Improved understanding of the genomic organization of different types of specialized metabolic pathways will shed light on the mechanisms underpinning pathway and genome evolution. It will also provide grist for the synthetic biology mill.

Modelling plant energy metabolism and economy: towards a synthetic biology to produce storage carbohydrate polymers

Oliver Ebenhöh[1,2]

[1]Institute for Complex Systems and Mathematical Biology, University of Aberdeen, Meston Building, King's College, Aberdeen, UK
[2]Cluster of Excellence on Plant Sciences (CEPLAS), Heinrich-Heine-University, Universitätsstraße 1, Düsseldorf 40225, Germany

Abstract

Most plants perform photosynthesis during the day and use the captured solar energy to reduce carbon dioxide to build sugars. These sugars are stored in the form of starch, an extremely compact and insoluble carbohydrate polymer. There are essentially two types of starch: storage starch, which is stored long term and is found e.g. in potato tubers, and transitory starch, which is stored in leaf cell chloroplasts and is regularly broken down during the night. Various aspects make studying starch metabolism from a theoretical perspective challenging. First, the turnover of transitory starch is extremely well timed, so as to avoid starvation near the end of the night while optimally using the stored reserves under a wide variety of conditions. Second, starch is a macroscopic and insoluble entity, therefore both starch synthesis and degradation processes involve enzymatic reactions taking place on the granule surface. These include polymer chain elongation and shortening, branching and debranching, as well as phosphorylation and dephosphorylation reactions. Further, at least during synthesis, biophysical processes play an important role in forming crystalline layers. In summary, carbohydrate storage metabolism in plants involves a variety of diverse processes, which are tightly controlled and timed. In order to reconstruct starch synthesis in a synthetic biology approach it will therefore be necessary to understand how the macroscopic structure of a starch granule emerges from the underlying microscopic biochemical and biophysical processes.

1 Introduction

Starch is a natural product produced by most land plants and algae with remarkable physico-chemical properties. Starch is composed of two polymers of glucose: amylose, a predominantly linear polymer of α-1,4 linked glucose units, and amylopectin, which also contains α -1,6 linkages (branch points) resulting in a tree-like structure [1]. The simple constituents of starch (one type of monomer and two types of linkages) is contrasted by its complex and highly ordered structure, in which crystalline and amorphous layers alternate in a defined and regular fashion. This structure gives starch unique physico-chemical properties, which make it an exceptionally tightly packed energy storage that is of such tremendous importance for the human diet and economy as a whole. Despite decades of intense research, it is still not understood how precisely starch granule biogenesis initiates and progresses. A relatively small number of enzymes are involved, but it is unclear how their activities are coordinated in order to ultimately control the structure and properties of starch.

From a socio-economic perspective, starch is extremely important. Not only is it the main calorific intake of humankind, but it also serves as a major bulk commodity for the chemical industry. Natural occurring starch displays a considerable variation in starch granule morphology, structure and composition between different botanic origins. All these factors are influencing starch properties, which are relevant for downstream functional applications, in particular in the light of further biotechnological and chemical applications. It would therefore be extremely useful if one could predict the emergent macroscopic physico-chemical properties of starch from the underlying microscopic biochemical and biophysical processes. Such ability would require reliable and realistic mathematical models, which can explain the emergent macroscopic properties from the underlying molecular mechanisms. However, the development of such models is extremely challenging due to a number of factors.

In the following we will discuss some of the remarkable aspects of starch metabolism and outline the reasons why they are difficult to treat from a theoretical perspective. The article will conclude with a perspective how synthetic biology approaches aiming at producing starch and controlling its properties may be approached.

2 Timing of Metabolism

In a multitude of experiments it has been demonstrated that plants of the species *Arabidopsis thaliana* have the remarkable capability to adapt their starch turnover rates to the photoperiod in such a way that at the end of the night almost all starch reserves, which were stored during the day, are consumed. Thus, the transiently stored starch is used in an optimal way, because the plant knows when dawn will arrive and energy and carbon can again be acquired by photosynthesis [2, 3]. More surprisingly even was the outcome of the pioneering experiment by Alexander Graf and coworkers, who imposed a sudden early night by placing plants, which were adapted to a 12:12 photoperiod, into darkness already after 8 hours of daylight. Remarkably, the plants immediately adjusted the starch breakdown rate to the lower starch content and longer dark period such that again almost all starch was consumed by the end of the night [4]. Understanding this enormous adaptability requires understanding various related phenomena. First, which regulatory mechanisms might allow the integration of cues from the circadian clock and the environment to arrive at such a fine-tuned regulation of starch synthesis and breakdown. Second, how can a plant determine how much starch it currently has stored and how can it determine how much time is left to dawn? Third, even if these quantities are somehow known to the plant, how does it perform the arithmetic division of these two quantities that is required to set the correct degradation rate? The latter of these problems was addressed by Scialdione and coworkers, who studied possible molecular interaction networks which can give rise to an arithmetic division algorithm [5] and could derive experimentally testable hypotheses on the nature of the molecular components performing the division. Even if the precise mechanisms plants use to perform the algorithmic division is not yet fully known, we at least understand how this task can in principle be achieved. This still leaves the problem how the two quantities, which need to be divided, i.e. starch content, S, and time-to-dawn, T, can be measured. For the time-to-dawn, there are various conceivable molecular networks, which would serve as internal timing mechanisms. In principle, they are all based on some coincidence timing mechanism, by which a regularly expressed clock output gene is used to 'reset' a molecular concentration (either by rapid degradation or by induced production) and a light cue (either presence or absence of light) will have a slower, but opposite effect on the molecule in question. This can easily result in a concentration which is proportional either to T or to $1/T$ [6, 7]. However, it remains mysterious how the plant is able to decide how much starch it currently has left. The difficulty here is that starch is insoluble and therefore osmotically neutral. In contrast to soluble substrates, it's amount is not trivially determining rates of enzymes using it as substrate – see below in Section 3. The fundamental question how a system with a regular input from a clock and a light-dependent stimulus must be designed in order to exhibit the observed accelerated starch synthesis in short days and accelerated starch breakdown in long days was addressed by a simplified model in [8], which provides a framework to understand the underlying principles behind this regulation.

Notwithstanding the gaps in our knowledge regarding this fundamentally important regulatory mechanism, mathematical models could provide considerable insight also into molecular mechanisms which might be used to implement such regulation. Based on a multitude of experimental observations, we could develop a more detailed model [9], in which we present specific candidates for the molecules playing the role of a molecular timer and an integrator of circadian and external cues.

3 Surface-active enzymes

Unlike most intermediates in cellular metabolism, starch is insoluble. Moreover, starch granules can have a huge size (depending on botanic origin, up to more than 50μm) [10]. Therefore, most of the starch is unavailable as a substrate for enzymatic or other chemical processes, because it is compactly stored and hidden beneath the starch granule's surface. Evidently, when developing mathematical models involving starch as a substrate or product, starch cannot simply be treated as any other soluble metabolite, because the reaction space is restricted to the granule's surface.

In contrast to classical enzymes acting on substrates in aqueous solutions, for which a comprehensive theoretical description was already developed by Michaelis and Menten over 100 years ago [11], a consistent treatment of surface-active enzyme is far less established [12, 13]. Often, the difficulty in providing a mechanistically correct and consistent description of enzymatic reactions on the starch surface is avoided by treating starch as an external metabolite [14, 15]. The reason why a more realistic treatment is often avoided is clearly the complexity of surface-active enzymatic processes. Once one starts to systematically investigate generic surface-active processes, one realises that a multitude of factors start to play a role which can safely be neglected in the case of soluble substrates. The simplest approach is to assume that enzymes adsorb and desorb from the reactant surface and perform some modification of the surface when they are in an adsorbed state [16]. This immediately poses the question which adsorption model should be used. Only the simplest, the Langmuir adsorption model [17], is analytically solvable. It assumes that the surface is composed of regular, non-overlapping binding sites. This is clearly a simplification, but already the still rather simple random sequential adsorption model makes finding analytic solutions for the binding equilibrium impossible [18]. Regardless which adsorption model is applied, assuming a rapid binding equilibrium leads to an overall rate equation which formally resembles the Michaelis-Menten equation known for the case in solution. However, there are a few remarkable differences between surface-active enzymes and their classical counterparts. Most importantly, the specific rate is – in contrast to enzymes acting in solution – no longer independent on the enzyme concentration. This can be explained by crowding effects: once the enzyme concentration is so high that a considerable part of the available surface area is occupied, many enzymes will be unbound and therefore catalytically inactive. Moreover, the apparent Michaelis constant is dependent on substrate properties as well as total enzyme concentrations. In particular, for particles with a high specific surface area (small particles), the Michaelis constant appears smaller, and higher enzyme concentrations leads to a higher apparent Michaelis constant [16].

4 Enzymatic reactions on polymers

A further difficulty when theoretically describing processes involved in starch synthesis or breakdown is the polymeric structure of starch and the pathway intermediates. Starch metabolism involves chain elongation, shortening, branching and debranching of carbohydrate polymers. Many of the involved enzymes are specific to a submolecular region, such as the non-reducing end of a glucan, but the specificity is often independent on the remaining part of the molecule. For example disproportionating enzyme 1 (DPE1), a glucanotransferase important in starch degradation, transfers a number (usually 2 or 3) glucosyl residues from one glucan to another, independent on their precise length [19]. This means that, theoretically, this enzyme catalyses an infinite number of reactions. Until recently, a comprehensive theoretical treatment of polymer systems was restricted to chemical systems [20, 21], which were already developed in the 1940's. In was not until a few years ago that we could illustrate how enzymatic systems on polymers can be described by concepts derived from statistical thermodynamics [22].

Essentially, different chain lengths (or degrees of polymerisation, DP) can be associated with different energy states, by describing a glucan of a certain DP by the total bond enthalpy contained in its interglucosidic linkages. In this framework, a polymer-active enzyme catalyses the transfer of particles between different energy states while observing certain constraints [23]. For example, the action of DPE1 can be described by raising one particle a number q of energy states up, while simultaneously lowering a particle the same number of states down. This analogy between biochemical systems on polymers and statistical thermodynamics opens tremendous new opportunities. Most directly, all formalisms developed for statistical physics can be directly applied to the biochemical systems, which allows for an easy calculation of the equilibrium states and furthermore provides an understanding which factors determine the equilibrium distribution and why. Conversely, since the biochemical systems are experimentally accessible through *in vitro* experiments, in which the temporal evolution of the DP distribution can in principle be monitored, there now exist experimental systems which may be used to test predictions from the still intensely researched field of non-equilibrium thermodynamics.

For practical applications to simulate starch-related pathways, stochastic simulations are a convenient compromise. For example, theoretical studies could illustrate and support the hypothesis how the involvement of polymer-active enzymes in the maltose consumption pathways leads to an increased robustness against fluctuations by implementing an entropy-driven metabolic buffer [22, 24].

5 Outlook: towards synthetic biology of starch

It is apparent that the processes of starch synthesis and breakdown and the mechanisms regulating these are highly complex and involve a number of different classes of factors. How then, and why, should one approach a synthetic biology project aiming at producing starch in a non-plant system?

There are two obvious answers to why one should embark on such an endeavour: First, there is the fundamental science aspect that we truly only understand a system if we are able to reconstruct it. Starch synthesis involves a diverse, but still rather limited number of enzymes, belonging to the three main classes of synthases, branching enzymes and debranching enzymes, and further involves non-enzymatic, biophysical processes. Still, the complexity is apparently too high to be understood by simple intuition. Therefore, a combined theoretical and experimental approach to reconstruct starch granules from scratch will result in a fundamental understanding how the macroscopic structure of a starch granule emerges from the basic microscopic processes. As such, understanding this emergence of the higher-ordered structure of a starch granule is a case study and a very first step to address the principle question in biology how living organisms emerge from the underlying microscopic processes. On a more practical side, starches of different physico-chemical properties are extremely important for dietary and industrial applications. In particular, differences in the branching patterns of amylopectin result in quite different properties of the granules. Being able to control these branching patterns and therefore the physico-chemical properties is invaluable for biotechnological applications. Again, a controlled biosynthesis of starch granules with desired properties seems only possible by a combined theoretical and experimental approach aiming at synthetically producing starch in a non-photosynthetic organism or even in a cell-free environment *in vitro*.

Agreeing that the reconstruction of a complex entity as starch is in itself an important and worthwhile enterprise and adopting the engineering view that understanding a system can only be achieved by the availability to build it *de novo*, lets us immediately derive priorities where one should start in a starch synthetic biology project. The regulatory mechanisms timing synthesis and degradation according to the day length are important and interesting in their own, but represent a higher level problem arising only once starch production is understood. Starch breakdown is also an interesting system, which in itself is difficult to understand. It does therefore make sense to try to investigate these two pathways separately. For this, an environment that usually does not produce starch is ideal. These can represent heterotrophic microorganisms such as *E. coli* or yeast, or even cell free *in vitro* systems. It appears a logic choice to start with starch synthesis because synthesis can run without degradation, while the reverse is not possible.

References

[1] S. C. Zeeman, S. M. Smith, and A. M. Smith, “The diurnal metabolism of leaf starch.,” *Biochem J*, vol. 401, pp. 13–28, Jan 2007.

[2] Y. Gibon, O. E. Bläsing, N. Palacios-Rojas, D. Pankovic, J. H. M. Hendriks, J. Fisahn, M. Höhne, M. Günther, and M. Stitt, “Adjustment of diurnal starch turnover to short days: depletion of sugar during the night leads to a temporary inhibition of carbohydrate utilization, accumulation of sugars and post-translational activation of adp-glucose pyrophosphorylase in the following light period.,” *Plant J*, vol. 39, pp. 847–862, Sep 2004.

[3] M. Stitt and S. C. Zeeman, “Starch turnover: pathways, regulation and role in growth.,” *Curr Opin Plant Biol*, vol. 15, pp. 282–292, Jun 2012.

[4] A. Graf, A. Schlereth, M. Stitt, and A. M. Smith, “Circadian control of carbohydrate availability for growth in arabidopsis plants at night.,” *Proc Natl Acad Sci U S A*, vol. 107, pp. 9458–9463, May 2010.

[5] A. Scialdone, S. T. Mugford, D. Feike, A. Skeffington, P. Borrill, A. Graf, A. M. Smith, and M. Howard, “Arabidopsis plants perform arithmetic division to prevent starvation at night.,” *Elife*, vol. 2, p. e00669, 2013.

[6] D. D. Seaton, O. EbenhÃ¶h, A. J. Millar, and A. Pokhilko, “Regulatory principles and experimental approaches to the circadian control of starch turnover.,” *J R Soc Interface*, vol. 11, p. 20130979, Feb 2014.

[7] A. Pokhilko, A. Flis, R. Sulpice, M. Stitt, and O. Ebenhoh, “Adjustment of carbon fluxes to light conditions regulates the daily turnover of starch in plants: a computational model,” *Mol. BioSyst.*, pp. –, 2014.

[8] F. G. Feugier and A. Satake, “Dynamical feedback between circadian clock and sucrose availability explains adaptive response of starch metabolism to various photoperiods.,” *Front Plant Sci*, vol. 3, p. 305, 2012.

[9] A. Pokhilko and O. Ebenhöh, “Mathematical modelling of diurnal regulation of carbohydrate allocation by osmo-related processes in plants,” *Journal of The Royal Society Interface*, vol. 12, no. 104, 2015.

[10] J. Fettke, L. Leifels, H. Brust, K. Herbst, and M. Steup, “Two carbon fluxes to reserve starch in potato (solanum tuberosum l.) tuber cells are closely interconnected but differently modulated by temperature.,” *J Exp Bot*, vol. 63, pp. 3011–3029, May 2012.

[11] L. Michaelis and M. Menten, “Kinetik der invertinwirkung,” *Biochem. Z.*, vol. 49, pp. 333–369, 1913.

[12] O. B. Berg and M. K. Jain, *Interfacial enzyme kinetics*. John Wiley & Sons, 2002.

[13] A. G. Marangoni, *Enzyme Kinetics: A Modern Approach.* Wiley, 2003.

[14] M. G. Poolman, D. A. Fell, and S. Thomas, “Modelling photosynthesis and its control.,” *J Exp Bot*, vol. 51 Spec No, pp. 319–328, Feb 2000.

[15] A. Nag, M. Lunacek, P. A. Graf, and C. H. Chang, “Kinetic modeling and exploratory numerical simulation of chloroplastic starch degradation.,” *BMC Syst Biol*, vol. 5, p. 94, 2011.

[16] O. Kartal and O. Ebenhöh, “A generic rate law for surface-active enzymes.,” *FEBS Lett*, vol. 587, pp. 2882–2890, Sep 2013.

[17] I. Langmuir, “The adsorption of gases on plane surfaces of glass, mica and platinum,” *J. Am. Chem. Soc.*, vol. 40, no. 9, pp. 1361–1403, 1918.

[18] J. Talbot, G. Tarjus, P. R. V. Tassel, and P. Viot, “From car parking to protein adsorption: an overview of sequential adsorption processes,” *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, vol. 165, no. 1-3, pp. 287 – 324, 2000.

[19] G. Jones and W. Whelan, “The action pattern of d-enzyme, a transmaltodextrinylase from potato,” *Carbohydrate Research*, vol. 9, no. 4, pp. 483 – 490, 1969.

[20] P. J. Flory, “Thermodynamics of heterogeneous polymers and their solutions,” *The Journal of Chemical Physics*, vol. 12, p. 425, 1944.

[21] A. V. Tobolsky, “Equilibrium distribution in sizes for linear polymer molecules,” *The Journal of Chemical Physics*, vol. 12, p. 402, 1944.

[22] O. Kartal, S. Mahlow, A. Skupin, and O. EbenhÃ¶h, “Carbohydrate-active enzymes exemplify entropic principles in metabolism.,” *Mol Syst Biol*, vol. 7, p. 542, 2011.

[23] O. Ebenhöh, A. Skupin, Ö. Kartal, S. Mahlow, and M. Steup, “Thermodynamic characterisation of carbohydrate-active enzymes,” in *Proceedings of the ”5th International ESCEC Symposium on Experimental Standard Conditions of Enzyme Characterizations”*, 2013.

[24] C. Ruzanski, J. Smirnova, M. Rejzek, D. Cockburn, H. L. Pedersen, M. Pike, W. G. T. Willats, B. Svensson, M. Steup, O. Ebenhöh, A. M. Smith, and R. A. Field, “A bacterial glucanotransferase can replace the complex maltose metabolism required for starch to sucrose conversion in leaves at night.,” *J Biol Chem*, vol. 288, pp. 28581–28598, Oct 2013.

Still Life, Pencils of Nature and Fingerprints: Biotech Art, Synthetic Biology and the new Green

Jens HAUSER[1]

[1] University of Copenhagen, DK

Abstract

Since the earliest anthropomorphic statues, myths of vivification surround artefacts made by the artist's hand. The animation of malleable matter stands in a long pictorial tradition, and from the 19th century, the biological metaphor is continued in the discussion of the artwork itself as an organism. By means of *form*, *material* or *process*, a touch of aliveness is staged, aiming at involving the viewer viscerally. Art has *imagined*, *represented* and *mimicked*, then simulated and – quite recently – *manipulated* living beings and systems for real. After painting, sculpture, automata etc., art in the late 20th century has employed "dry" informatics and robotics to stage aliveness, as well as, since shortly, "wet" cell and molecular biology. Transgenics, the synthesis of DNA sequences, so-called *biobricks*, molecular biological visual imaging media such as gel electrophoresis or DNA chips, cell and tissue engineering observable in real-time growth, the use of retroviruses and the cloning of bacterial plasmid DNA belong to the repertoire of a still marginal but resolutely experimental form of contemporary art today. Contemporary artists who enter the labs are particularly 'close to life', and the new discipline of Synthetic Biology is well suited to upgrade art historical paradigms. In parallel, the democratization of lab tools leads to their appropriation by tinkerers and tactical media activists who apply the critical potential of open source culture from the digital age of Media Art to grassroots DIY biology and biohacking.

Curator of the exhibition *assemble* | *standard* | *minimal*, Berlin 2015. (`http://foerderband.org/_data/150121__PI_Cohen_VanBalen_en.pdf`)

Genome evolution in synthetic yeast

Joel BADER[1]

[1] Johns Hopkins U., Baltimore, US

Abstract

The Saccharomyces cerevisiae 2.0 project (Sc2.0) aims to create a yeast cell with an entirely synthetic genome, with several fully synthetic chromosomes already complete. The synthetic genome has been designed to provide unique capabilities for exploring genome evolution: transposons, repetitive sequences, and other elements with questionable fitness benefit have been removed; tRNAs are being located to a synthetic neochromosome; and loxPsym sites, symmetric synthetic versions of loxP recombination sites recognized by Cre recombinase, have been added to permit rapid genome rearrangement through Synthetic Chromosome Recombination and Modification by LoxP-mediated Evolution (SCRaMbLE). We report on genome rearrangements observed in synthetic strains subjected to SCRaMbLE. Patterns of deletions are able to identify genes required for viability and fast growth. Analysis of inversions demonstrate that swapping 3' UTRs often has little functional consequence. Large duplications are also observed and are hypothesized to result from a double rolling circle mechanism relevant to plasmid copy number maintenance and to rearrangements in human cancer. Finally, we present a physical model for recombinations based on DNA looping.

Mitotic catastrophe: insights from synthetic yeast

Damien COUDREUSE[1]

[1] IGDR, U. Rennes, F

Abstract

Proper progression through the cell cycle is fundamental for cellular life. From the control of mitotic entry to the regulation of the onset of DNA replication, the cyclin-dependent kinase (CDK) family is the primary node of the circuit driving the eukaryotic cell division cycle. Modulation of CDK activity relies on a host of inputs, highlighting the complexity of this critical process. However, discerning essential controls from secondary regulation is a challenge that may limit our understanding of the core engine behind cell cycle progression. Building on past studies demonstrating that a single cyclin is sufficient for driving this process in fission yeast, we generate and analyze basic synthetic systems providing controlled CDK activity levels and bypassing a large part of the endogenous regulatory network.

This approach allows us to bring an alternative view of cell cycle control, suggesting a system whose central architecture may be simpler than expected. Interestingly, the simplicity of the synthetic yeasts we have built makes them particularly adapted for mathematical modeling and theoretical dissection of the organization of cell cycle control. I will show how a combination of modeling and experimental approaches using fission yeast cells operating with minimal cell cycle networks allowed us to propose a novel mechanism for the origin of mitotic catastrophe, a lethal result of major deregulation and overlap of cell cycle phases. Our studies highlight the importance of coupling classical genetics with synthetic biology and mathematical modeling for understanding normal and pathological cell cycle events.

Opportunities in enzyme discovery and engineering for synthetic biology

Magali REMAUD[1]

[1] LISBP / Toulouse White Biotech, F

Abstract

As key actors of biotransformation, enzymes can provide innovative solutions to develop sustainable processes and access to a large variety of bio-based molecules for food, feed, cosmetic, pharmaceutical, chemical or fine chemical industries. Nowadays, numerous approaches are available to discover and isolate novel enzymes from biological resources. In addition, we have now the ability to adapt the enzymes to desired applications and render then more specific, robust and well-adapted to specific usages. More and more sophisticated possibilities are also offered to design new enzymes that have no equivalent in nature.

Exploiting this natural or synthetic diversity is a real challenge for innovation. In the search for an appropriate catalyst for a given process, one first option is to explore and screen the existing biodiversity. To this end, one can turn to functional genomics or metagenomics supported by data mining and efficient screening protocols. Alternatively or in parallel, protein engineering can also be envisaged to tailor enzymes. Rational or semi-rational approaches combined with structural computational biology are indeed very efficient ways to enhance enzyme stability, change enzyme specificity in particular to optimize or build new metabolic pathways and deliver green synthetic tools meeting requested specifications.

These various approaches and their recent developments will be presented and discussed through illustrations issued from the recent achievements of the group of catalysis and enzyme molecular engineering of LISBP, Toulouse.

Elementary flux modes, minimal cut sets, and the design of optimal cell factories

Jürgen ZANGHELLINI[1]

[1] ACIB, Graz, AT

Abstract

Elementary flux modes (EFMs) are non-decomposable steady-state pathways in metabolic networks. They characterize phenotypes, quantify robustness or identify engineering targets. In fact, EFM analysis is ideally suited for metabolic engineering as it allows for an unbiased decomposition of metabolic networks in biologically meaningful pathways. By identifying desired and undesired network properties and using the concept of (constraint) minimal cut sets optimal production hosts can be designed. We will introduce the different approaches, highlight several applications, and discuss the limitations and possible exit strategies to overcome these limitations.

PART II ARTICLES

Systems pharmacology of levodopa absorption

Marouen Ben Guebila[1] and Ines Thiele*[1]

[1]Molecular systems physiology group, Luxembourg Center for Systems Biomedicine, University of Luxembourg, Campus Belval, Luxembourg.

Abstract

Parkinson's disease patients are recommended to follow either a low protein diet (LPD) or a redistributed protein diet (PRD) during levodopa treatment, due to a reported competition between amino acids and levodopa for intestinal absorption. PRD showed a better clinical outcome that can be the result of a synergestic functional interaction between amino acids and levodopa in intestinal antiporters. In order to study the complex interaction between both compounds, we have combined a whole body physiologically based pharmacokinetic (PBPK) model of levodopa with genome scale metabolic model of the small intestine enterocyte. We have identified the kinetic parameters (*e.g.* absorption, clearance) of the whole body model through curve fitting on levodopa pharmacokinetic data. The parameters were concordant with experimentally measured values reported in the literature. This approach will allow the generation of a mechanism-based hypothesis about the superiority of PRD over LPD and ultimately provide an evidence based augmenting diet for Parkinson's disease patients.

1 Introduction

Levodopa has been the gold standard treatment for Parkinson's disease for more than 40 years [1]. The biotransformation of the prodrug into dopamine in the brain allows to reverse the symptoms of Parkinson's disease. Its chemical structure is highly similar to aromatic amino acids (*e.g.* tyrosine is the synthesis precursor of levodopa). Both groups of metabolites share the same intestinal transporter for the absorption from the lumen and for the efflux into the portal vein. Consequently, levodopa compete with aromatic amino acids [2] for transport, which affects the clinical outcome of the treatment. Therefore, it is generally recommended either to lower the proteins in the diet (LPD) or to redistribute the daily allowance of proteins (PRD). Different studies reported a better clinical outcome with PRD [3]. It is hypothesized that dietary amino acids can improve the absorption of levodopa in specific cases.

*Corresponding author: ines.thiele@uni.lu

No systemic analysis of dietary amino acids uptake and levodopa absorption has been done yet. A computational modeling approach may help to formulate mechanism based diet recommendations (LPD vs PRD) for Parkinson's disease patients [4, 5]. Such computational modeling approach would require the consideration of the spatio-temporal relationship between levodopa and dietary amino acids in the small intestine, including their absorption, distribution, metabolism and excretion. Pharmacokinetic modeling could capture this aspect. Constraint based reconstruction and analysis (COBRA) methods allow to add genome-scale depiction of the biochemical pathways in the area of interest, which in our study is the gut wall.

The COBRA approach [6, 7] has been used for constructing manually curated, stoichiometry-based networks of biochemical reactions occurring in a defined biological system (*e.g.* metabolism of an organism). The reconstruction process has [8] starts from the genome annotation and the survey of corresponding biochemical reactions from the literature pertaining to the organism of interest. The condition-specific model is then obtained from the reconstruction by conversion into a mathematical format as following:

$$Reaction : A + B \rightarrow 2\,C$$

is converted to the stoichiometric matrix S

$$S_{m,n} = \begin{pmatrix} -1 & \cdots \\ -1 & \cdots \\ +2 & \cdots \end{pmatrix}$$

where the rows represent reactions and the columns represent metabolites. The same approach is applied to all the reactions in the system, which allows to obtain a matrix of reactions, metabolites, and their stoichiometric coefficients. After defining an objective function (*i.e.* a reaction that the system optimizes for, such as production of biomass). By applying different types of constraints (*e.g.* mass conservation, thermodynamics equalities, and inequalities), the possible solutions of the mathematical model are reduced (*i.e.* leading to a smaller set of possible system behaviors). These constraints allow to derive many different condition-specific models from one reconstruction [9]. These models can have different states, which can be translated biologically into different phenotypes (*e.g.* secretion of specific metabolites) [7].

The assembly of the human genome-scale metabolic reconstruction (RECON 1) [10] and its extended version (RECON 2) [11] allowed comprehensive modeling of human organ-specific metabolism (*e.g.* muscle, liver, enterocyte [12, 13, 14]).

RECON 2 includes 7440 reactions and 5063 metabolites involved in 354 metabolic functions [10]. The predictability of this metabolic network has been demonstrated, *e.g.* for the study of the inborn errors of metabolism (IEM) [12, 15]. Recently, the reconstruction of a small intestine epithelial cell (sIEC) [12] has been published and provided a deeper insight into the impact of diet and enzymopathies on the metabolism of the small intestine.

The major drawback of COBRA is that it does not permit to model metabolite concentrations and times series profiles as it assumes the modeled system to be in a steady state. Consequently, steady-state fluxes are computed that balance mass. This approach is also called flux balance analysis (FBA) [16]. Physiological ordinary differential equations (ODEs) based model can overcome this shortcoming. In particular, physiologically-based pharmacokinetics (PBPK) [17] whole body generic models have been formulated to describe the drug and metabolites distribution in the human body. These ODE models allow the representation of concentrations of compounds (*e.g.* xenobiotics) as a function of time. However, they require knowledge about many kinetic parameters. PBPK models have two groups of parameters, physiological parameters (*e.g.* volume of organs, blood flow through organs) and compound parameters related to the xenobiotic (*e.g.* permeability and dissolution parameters). The physiological parameters are taken from the literature, if available. Otherwise they can be identified from pharmacokinetic data using curve fitting techniques [17]. Usually, the unknown parameters are less than 10, which considerably decreases the complexity of the system. Combining these two approaches holds the promise that metabolic parameters can be computed using COBRA modeling, while concentration and time profiles can be captured by PBPK modeling.

In this study, we achieve such combination by coupling the absorption module of the PBPK model and the small intestine enterocyte metabolic model. The absorption module depicts the entire process of absorption with respect to seven compartments corresponding to seven anatomical segments of the intestine (from duodenum to the ileum). This module is also referred to as the Advanced Compartmental Absorption and Transit (ACAT) model [18]. These techniques will help to identify the dietary factors that are at the origin of levodopa fluctuations.

2 Materials and methods

2.1 Small intestine epithelial cell model

The small intestine epithelial cell (sIEC) model was obtained from [12]. The sIEC contains 433 metabolites taking part in 1282 reactions encoded by 611 genes. We added levodopa transport reactions as well as the genes encoding for these transporters according to the recently published experimental findings [2]. The transport of levodopa involves at the luminal level: an amino acid antiport encoded by SLC7A9 and SLC3A1 (Entrez gene ID 11136 and 6519);

and at the basolateral side: an amino acid antiport encoded by SLC7A8 and SLC3A2 (Entrez gene 23428 and 6520) and an aromatic amino acid uniport coded by SLC16A10 (Entrez gene ID 117247). The computation of the flux distribution is performed using FBA, using the COBRA toolbox [7]. The steady state assumption dictates that mass is conserved and the concentration of metabolites stays the same, thus

$$S.v = 0$$

where S is the $m{*}n$ stoichiometric matrix (with m metabolites and n reactions) and v represents the flux vector through each biochemical reaction. In addition, constraints are subjected to the model by limiting the lower (Vi, min) and upper (Vi, max) bound on each reaction, if known, such that

$$Vi, min \leq V(i) \leq Vi, max$$

If no experimental information on the bounds was available,
$(Vi, min) = 0\ mg/ml/hour$ and $(Vi, max) = 1000\ mg/ml/hour$ was defined for irreversible reactions and $(Vi, min) = -1000\ mg/ml/hour$ and and $(Vi, max) = 1000\ mg/ml/hour$ for reversible reactions.

2.2 Whole body physiologically based pharmacokinetics model of Levodopa

A generic whole body physiologically based kinetic model was implemented in order to describe the pharmacokinetics of levodopa. The human physiological values (*e.g.* blood flow, organ volumes) were fixed based on the literature [17]. The compound parameters were estimated through data fitting but constrained to stay close to the parameters that are reported in the literature from *in vitro* or *in vivo* experiments (Table 1).

Parameter	Model parameter	Literature value
Molecular weight	197.18 g/mol	197.18 g/mol [23]
Blood plasma coefficient	0.927	-
Elimination constant	5000 ml/hour	35000 ml/hour [25]
Log of permeability	-4.737	-2.39 [23]
Unbound fraction	0.65	0.6-0.9 [23]
Effective luminal intestinal permeability	1.72 cm/hour	1.22 cm/hour [24]
Transcription factor	7.3	-
Intestinal basolateral effective permeability	15.38	-

Table 1: Parameters obtained through curve fitting of levodopa kinetics on whole body PBPK model. Units are mentioned with the corresponding parameters and dimensionless otherwise.

The levodopa plasma concentrations were taken from a previous experiment [19] of therapeutic drug monitoring of 200 mg of levodopa of standard formulation with 50 mg of benserazide (*i.e.* a levodopa peripheral metabolism inhibitor) in healthy volunteers. The fit was performed using the fmincon algorithm with the GlobalSearch option implemented in Matlab (Matlab Release 2014b, The MathWorks, INC., Natick, Massachusetts, United States.). The goodness of fit was determined using the Kolmogorov-Smirnov test (99.94%) and visual inspection of the predicted values (Figure 2 - center).

2.3 Coupling both models

We coupled both models indirectly [20], which allowed us to dynamically link stoichiometric and kinetic models. This approach is extended up in scale as seven sIEC models are combined to the seven segments of the ACAT model (Figure 1).

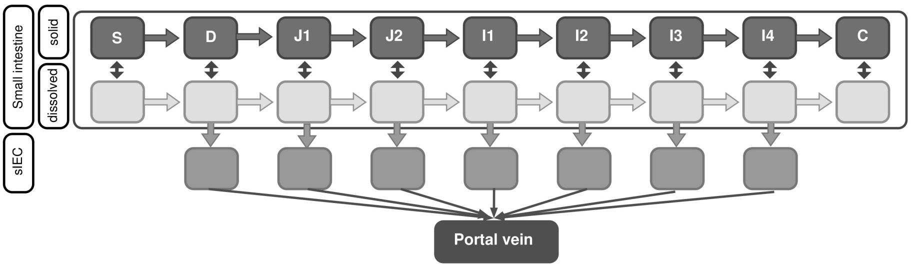


Figure 1: Combined compartmentalized absorption and transit model. The nutrients/drugs are modeled in different physical states (solid/dissolved in, respectively dark grey/light grey) throughout the stomach, small intestine, and colon. The sIEC models (medium gray) are combined to form the small intestine anatomical parts. S, D, J, I and C stand for stomach, duodenum, jejunum, ileum, and colon. The compartments are further segmented in the jejunum (1 and 2) and ileum (1, 2, 3 and 4).

The coupling is achieved in five steps [20]:

1. Divide the simulation time into steps and simulate ODE model for one time step.
2. Use the absorption rate as an upper bound for the COBRA model.
3. Simulate COBRA model, while levodopa luminal transport is the objective function.
4. Use obtained COBRA flux values and set them as rates for the ODE model.
5. Simulate ODE model for the next step .

3 Results

3.1 System identification of levodopa kinetics

Levodopa is a hydrophilic molecule. The active transport mechanisms happening at the site of action (*i.e.* brain) and elimination (*i.e.* kidney, liver) [21] allows for the distribution outside the blood, into the organs [22]. The kinetics of levodopa showed two phases of elimination from the body, consistent with the expected drug distribution in the peripheral tissue after the distribution in the central compartment (*i.e.* plasma) (Figure 2 - bottom). The obtained parameters are listed in table 1 and present a good compromise between data fitting and measured biological values. The blood plasma parititon (BPC) coefficient represents the fraction of levodopa that is absorbed by the red blood cell. The value is inferior to one, attesting a weak distribution into the red blood cells, which is expected for hydrophilic molecules. The overall elimination constant (Ke) is seven fold inferior to the reported parameter in the literature. In fact, the elimination of levodopa occurs through its transformation to dopamine, the elimination by the liver and by the kidneys. Modeling these elimination pathways did not improve the fitting (data not shown). These results underscore the fact that tissue specific elimination requires the measuring of levodopa concentrations in the eliminating organs. The effective luminal intestinal permeability (Peff) represents the rate of absorption of levodopa from the luminal side of the enterocyte and the intestinal basolateral effective permeability (BLeff) represents the secretion of levodopa from the basolateral side of the enterocyte into the portal vein. Although the latter parameter was not reported in the literature, it was demonstrated that the basolateral rate is superior to the luminal rate [18]. The identification of the kinetic parameters is the first step towards the coupling with the stoichiometric model of the enterocyte.

3.2 Coupled system

The indirect coupling of seven sIEC models to the corresponding anatomical segments was performed through an update loop that allowed for a feedback control between both types of models (*i.e.* kinetic and stoichiometric). When no additional constraints were imposed to the COBRA model, the combined model reached the upper bound and showed the same behavior as the kinetic model alone (Figure 2 - top). The addition of constraints, which represent the competition of amino acids and levodopa through a reduction of fluxes of the corresponding transporters, would reduce the secreted amount of levodopa in the systemic circulation, thus a lower concentration is expected.

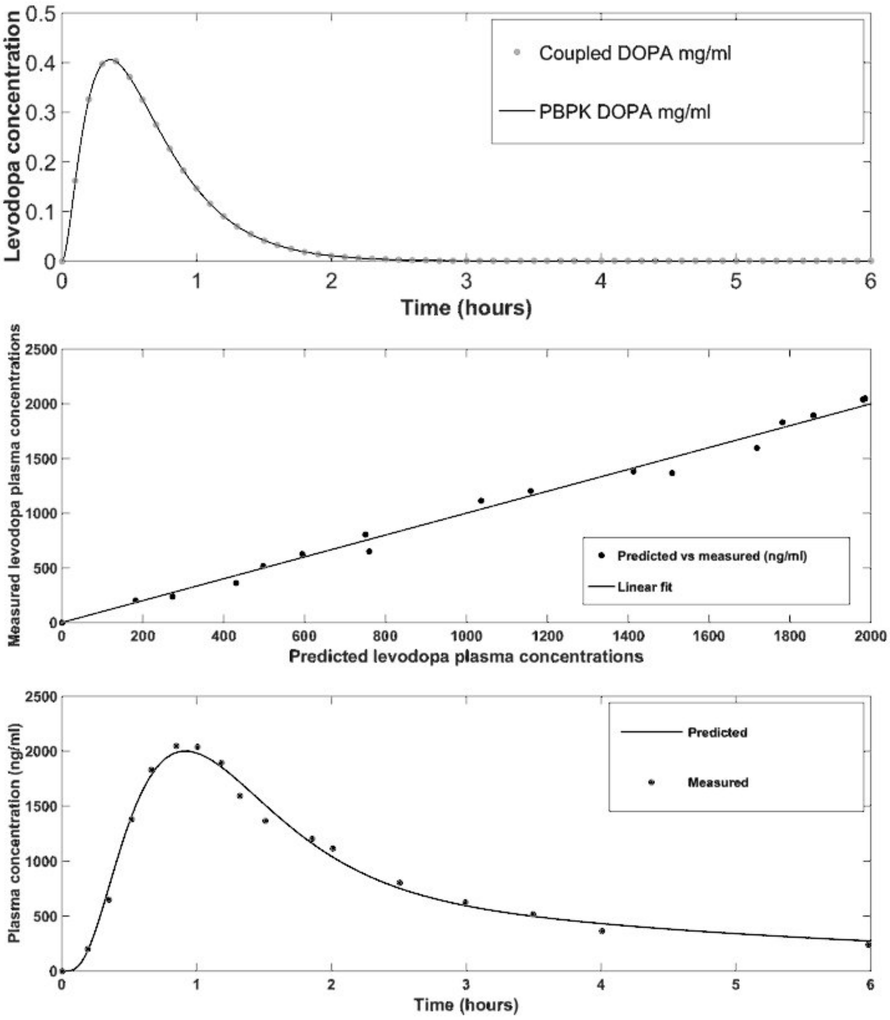


Figure 2: System identification of levodopa kinetics. Healthy volunteers levodopa plasma concentrations fit on whole body model (top) and goodness of fit plot (center) and the combined stoichiometric-kinetic model simulation without constraints (bottom).

4 Future perspectives

A recently published study identified the luminal and basolateral transporters of levodopa. These transporters are at the same time antiporters of dibasic and neutral amino acids. The presence of amino acids at the intracellular and extracellular space can interfere with the absorption of levodopa in many ways. Modeling the competition between levodopa and amino acids will allow to predict the pharmacokinetic profile of levodopa and identify the group of patients that needs dietary intervention (*e.g.* suppression of proteins in diet). Moreover, the presence of aminoacids in the portal vein representing the postprandial state, would enhance the absorption of levodopa *in vitro*. In this case, the coupled model could be used as a translational tool to assess the impact of this phenomenon in humans. The combined model will allow us to provide an evidence based hypothesis to clinically relevant observations such as the comparison of the outcomes of the low protein diet and the redistributed protein diet on levodopa kinetics and the optimization of diet composition for levodopa treated Parkinson's disease patients.

5 Acknowledgement

This work was supported by the University of Luxembourg.
We thank E. Bauer, A. Correia and A. Noronha for proofreading the manuscript.

References

[1] Contin, M. and P. Martinelli, Pharmacokinetics of levodopa. *J Neurol*, 2010. 257(Suppl 2): p. S253-61.

[2] Camargo, S.M., *et al.*, The molecular mechanism of intestinal levodopa absorption and its possible implications for the treatment of Parkinson's disease. *J Pharmacol Exp Ther*, 2014. 351(**1**): p. 114-23.

[3] Cereda, E., *et al.*, Low-protein and protein-redistribution diets for Parkinson's disease patients with motor fluctuations: a systematic review. *Mov Disord*, 2010. 25(**13**): p. 2021-34.

[4] Ishihara, L. and C. Brayne, A systematic review of nutritional risk factors of Parkinson's disease. *Nutr Res Rev*, 2005. 18(**2**): p. 259-82.

[5] Gao, X., *et al.*, Prospective study of dietary pattern and risk of Parkinson disease. *Am J Clin Nutr*, 2007. 86(**5**): p. 1486-94.

[6] Lewis, N.E., H. Nagarajan, and B.O. Palsson, Constraining the metabolic genotype-phenotype relationship using a phylogeny of in silico methods. *Nat Rev Microbiol*, 2012. 10(**4**): p. 291-305.

[7] Schellenberger, J., *et al.*, Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox v2.0. *Nat Protoc*, 2011. 6(**9**): p. 1290-307.

[8] Thiele, I. and B.O. Palsson, A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nat Protoc*, 2010. 5(**1**): p. 93-121.

[9] Beard, D.A., S.D. Liang, and H. Qian, Energy balance for analysis of complex metabolic networks. *Biophys J*, 2002. 83(**1**): p. 79-86.

[10] Duarte, N.C., *et al.*, Global reconstruction of the human metabolic network based on genomic and bibliomic data. *Proc Natl Acad Sci U S A*, 2007. 104(**6**): p. 1777-82.

[11] Thiele, I., *et al.*, A community-driven global reconstruction of human metabolism. *Nat Biotechnol*, 2013. 31(**5**): p. 419-25.

[12] Sahoo, S. and I. Thiele, Predicting the impact of diet and enzymopathies on human small intestinal epithelial cells. *Hum Mol Genet*, 2013. 22(**13**): p. 2705-22.

[13] Gille, C., *et al.*, HepatoNet1: a comprehensive metabolic reconstruction of the human hepatocyte for the analysis of liver physiology. *Mol Syst Biol*, 2010. 6: p. 411.

[14] Nogiec, C.D. and S. Kasif, To supplement or not to supplement: a metabolic network framework for human nutritional supplements. *PLoS One*, 2013. 8(**8**): p. e68751.

[15] Sahoo, S., *et al.*, A compendium of inborn errors of metabolism mapped onto the human metabolic network. *Mol Biosyst*, 2012. 8(**10**): p. 2545-58.

[16] Orth, J.D., I. Thiele, and B.O. Palsson, What is flux balance analysis? *Nat Biotechnol*, 2010. 28(**3**): p. 245-8.

[17] Jones, H. and K. Rowland-Yeo, Basic concepts in physiologically based pharmacokinetic modeling in drug discovery and development. *CPT Pharmacometrics Syst Pharmacol*, 2013. 2: p. e63.

[18] Agoram, B., W.S. Woltosz, and M.B. Bolger, Predicting the impact of physiological and biochemical processes on oral drug bioavailability. *Adv Drug Deliv Rev*, 2001. 50 Suppl 1: p. S41-67.

[19] Keller, G.A., *et al.*, Comparative bioavailability of 2 tablet formulations of levodopa/benserazide in healthy, fasting volunteers: a single-dose, randomized-sequence, open-label crossover study. *Clin Ther*, 2011. 33(**4**): p. 500-10.

[20] Krauss, M., *et al.*, Integrating cellular metabolism into a multiscale whole-body model. *PLoS Comput Biol*, 2012. 8(**10**): p. e1002750.

[21] Felder, R.A. and P.A. Jose, Mechanisms of disease: the role of GRK4 in the etiology of essential hypertension and salt sensitivity. *Nat Clin Pract Nephrol*, 2006. 2(**11**): p. 637-50.

[22] Meiser, J., D. Weindl, and K. Hiller, Complexity of dopamine metabolism. *Cell Commun Signal*, 2013. 11(**1**): p. 34.

[23] `http://www.drugbank.ca/drugs/DB01235`

[24] Lennernas, H., *et al.*, The effect of L-leucine on the absorption of levodopa, studied by regional jejunal perfusion in man. *Br J Clin Pharmacol*, 1993. 35(**3**): p. 243-50.

[25] Martinelli, P., *et al.*, Levodopa pharmacokinetics and dyskinesias: are there sex-related differences? *Neurol Sci*, 2003. 24(**3**): p. 192-3.

Extracting logic gates from a metabolic network

Marc Bouffard[1], Franck Molina[2] and Patrick Amar*[1,2]

[1]LRI, Université Paris Sud - UMR CNRS 8623, Bât. 650, F-91405 Orsay Cedex
[2]Sys2diag, FRE CNRS 3690, 1682 rue de la Valsière, F-34184 Montpellier

Abstract

In this article we will show how synthetic biology can be used to design artificial bio-devices that can perform successively the samples intake, its analysis and provide an integrated response.

The main part of our study is focused on the analysis task. This task will be performed by a logic function applied to the output of the sensors that respond to the bio-markers of the targeted pathology. This logic function is computed using logic circuits made of interconnected logic gates.

For this purpose, a library of basic logic gates that can be wired together so as to function correctly in the same environment, will be automatically extracted from the metabolic networks of living organisms.

1 Introduction

Depending on the pathology, traditional medical diagnostic is often hard to perform or invasive for the patient, or need heavy equipments. We want to use synthetic biology to design diagnostic devices that can be easy to use, cheap and non invasive.

Our goal is to design artificial biosystems that can sense the presence, or the absence of some of the biomarkers of a specific human pathology, and give back an easy-to-read response (e.g. colorimetric) [7]. These *bio-computers* could be used mixed with blood or saliva, or urine samples as a quick and non invasive medical diagnosic system. In a futuristic view, they could even be directly absorbed by the patient to diagnose some pathologies of the digestive system.

These bio-devices have to include material to perform three different types of functions: (i) biosensors to detect the biomarkers, (ii) a computing system to integrate the response of the biosensors and (iii) a display system to show the results. In this article we show how to build the logic circuits that are part of the computing system.

The use of synthetic biology to build logic circuits is not that recent [5]. Several authors [3, 9] have already shown that it was possible to build molecular logic gates, and even to chain them [6, 8]. But unfortunately, the molecular

*corresponding author: pa@lri.fr

logic gates shown in these studies use specific properties of some molecules which make them hard to wire together and therefore not very easy to use to build large circuits.

In our approach, a logic gate is implemented by a set of reactions where the inputs are the metabolites that are the substrates and the output is the product of a cascade of reactions catalysed by enzymes. We will show how to assemble independent functional units, the biochemical logic gates, to get large biochemical logic circuits. To maximise the efficiency of our logic gates, we will try to make them as little dependent as possible on the enzymes kinetics.

To avoid erratic behaviours that may be due to the stochasticity or to high local concentrations, we use a large enough number of copies of enzymes and metabolites to insure that the mixture is well stirred.

2 Biochemical logic gates

We wish to design logic micro-systems using biochemical reactions between various molecular species. To obtain a full circuit implementing a given boolean function, we need to design logic gates that can be connected together. Our logic gates will take as input a set of signals representing the boolean values *true* and *false*, then process them to output the same kind of signal. So we need to translate the boolean information in a type of signal that can be processed by biochemical reactions. To do this we use specific molecular species, metabolites, to implement the value of the boolean information at each point of the circuit (i.e.: the equivalent of the *wire* that connects the output of a gate to the input of the next one in electronic circuits).

A given metabolite will represent the boolean constant *false* if its average concentration in the vesicle is below a predefined threshold, th_1. Conversely, the boolean constant *true* will be represented by a concentration above another predefined threshold, $th_2 \geq th_1$. We assumed that if, at the beginning the computation of a logic function, the concentration of each input metabolites is not between the two thresholds, even if during the transient phase of computation, some of the metabolites concentrations may lie between th_1 and th_2, at the end, the concentrations of the output metabolites are either below th_1 or above th_2.

Using this representation of boolean values, we will need as many different molecular species as the number of connections needed to build the circuit. For example, a logic gate with two inputs and one output, will need three different molecular species; To connect its output to one of the inputs of the next gate, this gate must use the same molecular species for this input.

We define the biochemical logic gates by their truth table, the set of molecular species representing the inputs, the molecular species representing the output, and the metabolic network implementing the gate.

A gate is made of a set of enzymes that catalyse reactions between metabolites. A complex gate is implemented by a cascade of enzymatic reactions. A gate is starting to work when at least a few of its inputs become available to the first reaction. Then the next reactions can begin, producing new metabolites while consuming the ones produced by the previous reactions in the cascade. Finally, the gate begins to produce its output when the last reaction has its substrates available.

The number of layers (i.e. successive reactions) needed to implement a particular gate may vary, depending on its complexity and on its number of inputs. If a logic gate is complex enough, many different enzymatic networks can be used to implement it, which can lead to different numbers of layers.

2.1 Simple Gates

A simple logic gate, with two inputs and one output, can be built with only one layer:

- AND Gate: we use a reaction catalysed by an enzyme E that needs two substrates, A and B (the inputs) to produce a metabolite, C (the output) (fig. 1 left).

- OR Gate: we use two reactions that produces the same metabolite, C (the output) one from a substrate A (first input) the other from a substrate B (second input). We can use one (or more probably two different) enzymes, E1 that catalyse A → C and E2 that catalyse B → C (fig. 1 right).

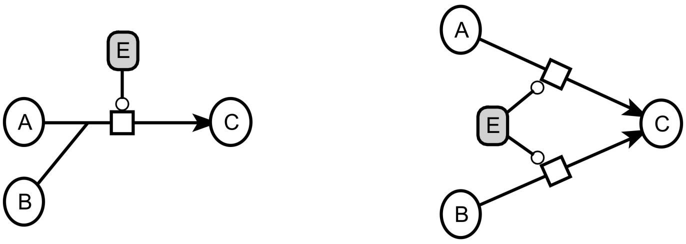


Figure 1: Simple networks for two inputs AND (left) and OR (right) biochemical logic gates.

These logic gates are made of one or two enzymes in only one layer. An important characteristic of these gates is that they do not depend on the kinetics of the enzymes to compute the correct boolean value. Furthermore, the two inputs of the gate do not need to be strongly synchronised, the gate is correctly working regardless the order of arrival of the input values.

For the inverter (NO gate) or when one of the inputs must be inverted or if it is the output that must be inverted (NAND, NOR gates) the main problem is to be able to react to the absence (or low concentration) of a metabolite.

To address this problem, we have used two different methods (i) inhibition and (ii) competition. Here is an example of two versions of the $C = \bar{A} \wedge B$ gate based on each of these principles. In both versions, the low concentration of the input A metabolite leads to a concentration of the output product that follows the concentration of the input B metabolite:

- $C = \bar{A} \wedge B$ gate using inhibition: we use an enzyme E that catalyses the reaction B → C, where metabolite B is one input, and C is the output product. This enzyme E being inhibited by the second input metabolite A (fig. 2 left).
- $C = \bar{A} \wedge B$ gate using competition: we use an enzyme E_1 that catalyses the reaction $\mathrm{R}_1 : \mathrm{B} \rightarrow \mathrm{C}$. Another reaction, R_2, catalysed by E_2 using both the input metabolites A and B to produce an unused metabolite P, is running concurrently. When metabolite B is present, the presence of metabolite A leads to the consumption of B by the reaction R_2, such that reaction R_1 is not very active, so the output metabolite C has a low concentration (fig. 2 right).

A common drawback to both of these methods is that they are heavily dependent on the relative kinetics of the reactions. For the inhibition method, it is crucial that the inhibition is stronger than the reaction B → C. For the competition method, reaction R_2 must have a higher kinetics than reaction R_1.

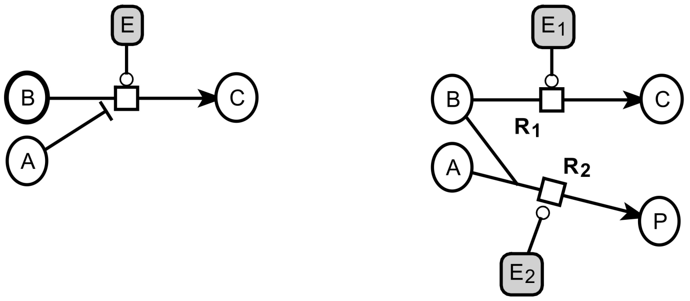


Figure 2: $C = \bar{A} \wedge B$ gate using inhibition (left), or competition (right)

2.2 Circuit wiring

In order to wire gates together, the molecular species representing the output of a gate must be the same as the one used for an input of the following gate, the next layer (fig. 3).

This is a strong constraint that must be satisfied by the implementation of the gates in order to get the desired circuit.

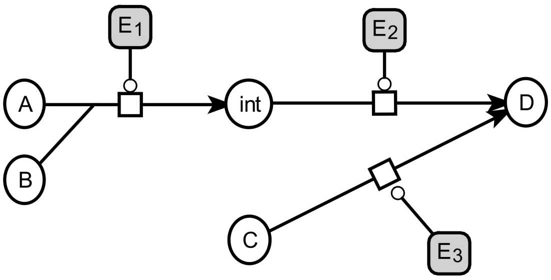


Figure 3: Circuit computing $D = (A \wedge B) \vee C$; The product int of enzyme E_1 must be the same as the substrate of enzyme E_2 in order to wire the output of the AND gate to on input of the following OR gate

The most important difference between the metabolic and the electronic implementations of a circuit computing a given boolean function lies in the way the wiring is made. The electronic gates use wires, insulated from each others, according to a permanent connection scheme. In an electronic circuit, all the instances of the same kind of gate are identical since they are connected using non cross talking wires.

Our biochemical implementation does not benefit from this clear separation between each basic gate. The *wires* we use are insulated because they are not made of the same material (metabolic species), therefore they can reside in the same environment without any cross talk because of their very nature (fig. 3). There are two consequences to this: (i) each occurrence of a given kind of gate must be unique to avoid short circuits and (ii) the connection scheme is not permanent since a connection begins to exist (i.e.: carrying the *true* boolean value) when the concentration of the corresponding molecular species raises above a threshold, and disappear when its concentration decreases under another threshold.

In our approach, the metabolites that implement the connections are created, processed and consumed, so they have a limited lifetime for most. We can consider that when after a while some metabolites are consumed, the corresponding wires no longer exist. The logic gate implemented by a biochemical network is in some way built only at the moment the metabolites representing its inputs are present, building (i.e. activating) each layer of reactions until the final one. When all the input and internal metabolites are consumed, the gate is *deconstructed*, each enzyme that is part of this gate could be even reused to build another (type of) gate.

3 NetGate: from a metabolic network to a set of logic gates

There are many environmental factors (temperature, pH, etc.) that may have an influence on the functioning and on the kinetics of the reactions catalysed by enzymes. Since our ultimate goal is to build biochemical nano-systems using enzymatic reactions that take place in the same environment (lipidic vesicles or droplets), it is best to use enzymes and metabolites that are already part of the same metabolic network in the same environment *in vivo*. Hence the idea to extract as many different logic gates we can from an existing metabolic network.

Before describing in details the algorithms implemented by NetGate, let's set some definitions that will help to understand them:

1. A *metabolic network* is a set of interconnected reactions involving metabolites (substrates or products) and enzymes (catalysts or modulators). They form a dense and usually strongly connected network.
2. *Tied reactions*: two reactions are tied if they share at least one common molecular species.
3. A *logic gate* is an abstract construction with at least one input and one output. A truth table is associated to the gate; The truth table gives the value of each output for each possible boolean pattern of the inputs. The set of logic gates NetGate is searching for are described by their truth table in a parameter file.
4. An *implementation* of a logic gate is a subnetwork of the input metabolic network where the inputs and output are identified. The number of inputs of the subnetwork may exceed the number of inputs of the gate. If the value of one of these extra-input does not change the behaviour of the gate, this input is left free and will be ignored. Conversely, if any variation of the value of an extra-input changes the behaviour of the gate, then a fixed boolean value is assigned to this input in order to get the correct truth table for the gate.

3.1 Overview

NetGate takes as inputs (i) a SBML file describing a metabolic network and (ii) a list of truth tables corresponding to the logic gates that are to be searched in the metabolic network.

First, all the possible implementations of the logic gates are enumerated; Then, these implementations are checked against the given list of truth tables and the gates found are sorted and output.

The gates implementations are searched in subnetworks extracted from the original metabolic network. These subnetworks are built starting from one reaction of the original network, the seed, then adding successively other reactions that are tied to this seed. To get all the subnetworks, this process is repeated starting from all the reactions of the original metabolic network.

Then, for each of the given gate description, all the possible implementations are searched within each subnetwork. All the mappings of the inputs of the gate to the inputs of the subnetwork are successively checked to see if all the lines of truth table of the gate description can be obtained.

3.2 The algorithm in-depth

3.2.1 Metabolic network conversion

The input metabolic network is translated to a reaction matrix, where the lines are the reactions and the columns the molecular species (metabolites and enzymes). As we assume a stoichiometry of 1, we use -1 for a reactant that will be consumed, +1 for a reactant that will be produced, and 0 when a reactant is neither produced nor consumed. The network on fig. 4 will be translated to the reaction matrix on fig. 5.

All the metabolic reactions are reversible, but depending on the enzyme that catalyses the reaction, the equilibrium can be unbalanced; In this case, the reaction is considered irreversible. The reversible reactions are split into two reactions: the forward reaction and the reverse reaction. An artificial enzyme will be used to catalyse the reverse reaction while the real enzyme will catalyse the forward reaction.

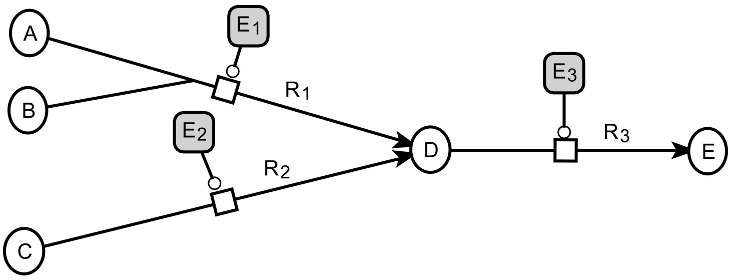


Figure 4: A simple metabolic network.

Reaction #	E_1	E_2	E_3	A	B	C	D	E
R_1	Cat	0	0	-1	-1	0	+1	0
R_2	0	Cat	0	0	0	-1	+1	0
R_3	0	0	Cat	0	0	0	-1	+1

Figure 5: Reaction matrix of the network of fig. 4. (*Cat* means that the enzyme is needed to catalyse the reaction)

3.2.2 Decomposition in subnetworks

To enumerate all the subnetworks of the original metabolic network a tree-like greedy algorithm on the original network will be used:

1. start from one reaction of the metabolic network to constitute the first size one subnetwork.
2. build a new subnetwork by aggregating to the current subnetwork one reaction taken from the original network, which is tied to one of the reactions of the current subnetwork. The added reaction must not be the opposite reaction of an already included reversible reaction (i.e. the forward or the reverse reaction may be part of the subnetwork but not both).
3. repeat step 2 until no new reaction can be added.

To build all the subnetworks, this algorithm is re-applied on all the reactions on the metabolic network, using each time a different initial reaction. It is very probable that the same subnetwork appears more than once, so the algorithm actually used has been optimised to cut down the exploration tree starting from a subnetwork that has already been explored. Eventually, this process enumerates all the different subnetworks of the initial metabolic network.

3.2.3 Finding the gates implementations

At this point of the process, for each gate we are interested in, we will look for all the implementations we can find in each of the subnetworks built at the previous step.

There is a huge number of *potential implementations* because frequently, a logic gate has a small number of inputs compared to the number of inputs of the average subnetwork. The naive way to find all the potential implementations for a logic gate with n inputs and one output that can be extracted from a subnetwork with $k \geq n$ inputs and $m \geq 1$ outputs, is for each of the m outputs, to enumerate the number of combinations of n among k (fig. 6). Therefore we can obtain up to $m \cdot \binom{n}{k}$ potential implementations, but some of them may be clearly discarted if one of the inputs has no effect on the output. Moreover, many of these potential implementations may be identical if one (or more) of the $k - n$ extra-inputs also have no effect on the output.

We can optimise the search by computing for each output of the subnetwork, the list of its *predecessors*: the molecular species that are inputs of the subnetwork, which may influence the output. Then a set of potential inputs associated to a given output is made using the predecessors of this output. Each one of these sets of reactions constitutes a potential implementation for a logic gate.

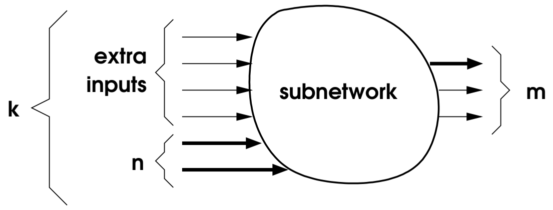


Figure 6: Subnetworks with k inputs and m outputs. One of the $m \cdot \binom{n}{k}$ possible logic gate implementation with $n = 2$ inputs and one output is shown with thick arrows. For each of the logic gate searched, a coarse grained simulation is made in order to see if there is a match with each line of its truth table. This is done for all the binary combinations of the extra-inputs.

3.2.4 Testing the gates implementations

Overview of the test procedure

Once we have a potential implementation, we compare it to each logic gate that we are interested in, in order to find a possible match. To do this, we evaluate the output of the implementation (the reactions subnetwork) for all the possible boolean configurations of the inputs in order to find if one of these evaluations matches the truth table of a specific logic gate.

To evaluate a line of a truth table, firstly we initialise the concentrations of the input metabolites with values corresponding to the boolean values of that line. Then, the other extra-inputs of the subnetwork are successively set to the metabolites concentrations that correspond to all their possible binary configurations. Finally, a coarse-grained simulation of the dynamics of the reaction network is applied in order to give us an approximation of the concentration of the output metabolite, and therefore an approximation of its binary value.

This process is performed for all the lines of the truth table of each logic gate that are searched, and for all the valuations of the extra-inputs in order to find if *this logic gate* can be implemented by *this reaction network* according to *these values* of the extra-inputs.

Coarse-grained simulation algorithm

The simulaton of the dynamics of the network is very similar to the simulation of a Petri net: the places are the molecular species and the transitions are the reactions. An initial valuation of all the places is made and then a certain number of simulation steps are performed. The update scheme used is deterministic (each time a transition can be fired, it is fired) and synchronous (all the fireable transitions are fired at the same time).

Here, the tokens represent the concentrations of each molecular species, and the transitions are computed directly from the reaction matrix. Our simulation system is coarse-grained in the sense that since we are primarily interested in a boolean behaviour, the concentrations of the metabolites (and consequently of the enzymes) are represented by a few tokens. Initially, the places corresponding to inputs set to the boolean value *false* have 0 tokens, and those set to *true* have 10 tokens. The extra-inputs initially set to *true* have a high value of 1000 tokens. All the places corresponding to enzymes have initially 2 tokens. The inhibited reactions are represented by the artificial consumption by the inhibitor of the enzymes catalysing the reactions.
Conversely, at the end of the simulation, the number of tokens in the places representing metabolites are converted to the boolean value *false* when this number is less than 4, and to *true* otherwise.

Using these conventions, the boolean values of the inputs for each line of the truth table of the currently tested gate are converted to the corresponding initial number of tokens in each place, and the simulation process may begin.

Time index	E_1	E_3	A	B	D	E	Reactions
0	2	2	10	10	0	0	
1	2	2	9	9	1	0	R_1
2	2	2	8	8	1	1	R_1 & R_2
3	2	2	7	7	1	2	R_1 & R_2
...	...	...	...	...	...	...	...
9	2	2	1	1	1	8	R_1 & R_2
10	2	2	0	0	1	9	R_2
11	2	2	0	0	0	10	

Figure 7: Coarse-grained simulation of the subnetwork of fig. 4 made of only reactions R_1 and R_3. This simulation shows that when inputs A and B are set to the boolean value *true*, after a certain amout of time, the output E switches from *false* to *true* (which is a good start for an AND gate).

The simulation process is quite simple: at each iteration, all the fireable transitions are triggered and the number of tokens in each place is updated. This process continues until either there are no more fireable transitions, or a predetermined maximal number of iterations is reached (to avoid infinite cycles when there is a loop in the reaction network).

A reaction is triggered when all its substrates (-1 in the reaction matrix) are present and the enzyme that catalyses the reaction is also present. All the reactions are triggered synchronously, therefore to avoid negative numbers of tokens, if there are not enough tokens in some places, the reactions using these places are not triggered (fig. 7).

We can see a token as a unit of metabolite quantity of matter. The value of time slice associated to each iteration is not defined and we do not know the kinetics of the reactions. But nevertheless, this coarse-grained computing model allows us to have a good idea of the global dynamics of a small set of reactions, and therefore can predict with a good accuracy if this set of reactions can or cannot implement a given logic gate.

3.2.5 Sorting and storing the gates

It is highly probable to find multiple versions of the same logic gate, implemented by the same set of reactions but extracted from different subnetworks. For example, a specific logic gate can be found in a subnetwork SN_1 of size 3 (made of 3 reactions), and also in another subnetwork SN_2 of size 4, which is just an extended version of SN_1.

It is also possible to find multiple versions of the same model of logic gate (i.e. same truth table, and same molecular species for the inputs, and the output) but implemented by different sets of reactions, and different sets of extra-inputs (and/or valuations for them).

To avoid these kinds of useless redundancies, we will keep only the simplest version of each gate. The simplest version of a gate is the one implemented by (i) the smallest set of reactions, and (ii) the smallest number of extra-inputs.

If a new implementation of an already stored gate is found, and if the new implementation has the same number of extra-inputs, and the same number of reactions but if at least one reaction is different, this new implementation is also kept (it is a good alternate version of the same gate).

This last part of NetGate is implemented using association lists that allows the validation subroutine to quickly find whether the new found gate implementation is to be discarted or must replace or is to be added to the already found set of implementations of the same logic gate.

3.3 Validation and results

NetGate is written in the C++ programming language. It has been compiled using *gcc* on the Linux, MacOSX and Windows 7 operating systems on 32 bits and 64 bits architectures. The program is parallelised, using one specific thread per type of logic gate searched. The default set includes 7 of the most commonly used two-input logic gates.

In less than 16 seconds, on a quad-core Intel I5 Apple MacBook Air computer, NetGate has found 1759 logic gates with at most 4 reactions per gate, in a 35 enzymes version of the *B. subtilis* central carbon metabolism. During its execution, the program has created 820 subnetworks, evaluated 2,155,496 potential logic gates and has found 836 AND and 923 OR gates.

Many thousands implementations of the logic gates found by NetGate have been verified using the HSIM stochastic simulation system [1, 2]. A specific format converter have been written to automatically translate the output of NetGate to input models for HSIM.
These models share these common characteristics:

- the reactions take place in a virtual spherical vesicle of 0.4 μm of diameter (volume $\approx$ 0.0335 μm^3).
- The initial concentrations of the metabolites for all the inputs having the *true* boolean value are set to $\approx$ 1 mM (20,000 copies).
- The concentration of the enzymes is set to $\approx$ 2.5 μM (50 copies).
- The threshold used to determine a boolean *true* value is $\approx$ 0.5 mM (10,000 copies).

All the simulations showed that the logic gates implementations found by NetGate reproduced correctly their truth tables.

To test extensively NetGate we used a dataset downloaded from the KEGG pathway database [4]. This dataset includes 72,095 different metabolic networks coming from various living organisms. The sizes of the metabolic networks varies from one to eighty reactions. These validation tests have been made only on the metabolic networks that have more than five reactions.

4 Conclusion

NetGate proved useful to create a library of logic gates that can be interconnected together and that can function correctly when located in the same environment. The validation tests have shown that our model of biochemical logic gates were robust and very well adapted to our purpose.

The next step of the computer aided design of diagnosis bio-devices is to be able to automatically propose various biochemical implementations of a given boolean function. Another step is to also propose pre-built circuits that implement complex functions, such as timers, oscillators, memories, finite state machines, etc.

References

[1] Amar, P., Legent, G., Thellier, M., Ripoll, C., Bernot,G., Nystrom, T., Saier, M. and Norris, V. 2008. *A stochastic automaton shows how enzyme assemblies may contribute to metabolic efficiency*. BMC Systems Biology, 2(27).

[2] Amar, P. and Paulevé, L. 2012. *HSIM: an hybrid stochastic simulation system for systems biology* In ENTCS, The Third International Workshop on Static Analysis and Systems Biology, SASB 2012.

[3] Bray, D. 1995. *Protein molecules as computing with proteins*. Nature vol. 376, 307–312

[4] *KEGG PATHWAY Database*.
`http://www.genome.jp/kegg/pathway.html`

[5] Marguet, P., Balagadde, F., Tan C. and You, L. 2007. *Biology by design: reduction and synthesis of cellular components and behaviour*. J. R. Soc. vol. 4, nr. 15, 607-623, Rsif.royalsocietypublishing.org

[6] Niazov, T. Baron, R. Katz, E. Lioubashevski, O. Willner, I. 2006. *Concatenated logic gates using four coupled biocatalysts operating in series*. PNAS vol. 103 nr. 46, 17160–17163

[7] Rialle, S., Felicori, L., Dias-Lopes, C., Peres, S., El Atia, S., Thierry, A., Amar, P. and Molina, F. 2010. *Bionetcad: design, simulation and experimental validation of synthetic biochemical networks*. Bioinformatics, 26(18):2298–2304

[8] G. Strack, M. Ornatska, M. Pita, E. Katz. 2007. *Biocomputing Security System: Concatenated Enzyme-Based Logic Gates Operating as a Biomolecular Keypad Lock*, J. Am. Chem. Soc. vol. 130, nr. 13, 2008 4235

[9] Unger, R. and Moult, J. 2006. *Towards computing with proteins*. Simulation. 63, 53–64

Probabilistic Gene Network

Kristine Joy E. Carpio[1,3], Gilles Bernot[2],
Jean-Paul Comet[2] and Francine Diener[1]

[1] Laboratoire J.A. Dieudonné, Université de Nice - Sophia Antipolis, France
[2] CNRS UMR 7271, Laboratoire I3S, Université de Nice - Sophia Antipolis, France

Abstract

In this article we present a modelling framework that links the well known modelling framework of gene network introduced by R. Thomas and Markov chains. In a first development we introduce a Markov chain having as state space the set of all possible states of the R. Thomas models: we generate the transition probabilities by examining all the possible parameterizations of the interaction graph. The second development focuses on a stochastic framework where several parameterizations of a same qualitative gene interaction graph are considered and transition probabilities allow one to jump from a state to another one which can potentially be in another parameterized model. The idea is to consider only parameterized qualitative models of R. Thomas which abstract biological knowledge, and to use transition probabilities to allow to jump from one to another, if information coming from biological experiments reinforces the belief in a particular model.

1 Introduction

Regulatory networks are models based on graphs which are used to obtain a simpler view of gene regulation [6, p. 101]. Gene regulation is defined as the process of turning genes on and off which is made possible by a network of interactions that includes chemically modifying genes and using regulatory proteins. Gene regulation guarantees that appropriate genes are expressed at proper times specially during early development where cells begin to take on specific functions; it also helps an organism respond to its environment [8].

The different frameworks for modelling gene networks can be classified into three main groups. The systems of differential equations have been largely used in order to represent a lot of systems with a lot of details (transcription, traduction, transports ...). The second group consists of stochastic frameworks like Markov chains. The Markov modelling framework supposes that, given the past and the present, the future only depends on the present [9, p. 163].

[3] On research leave from De La Salle University, Philippines. Funded by Emma in the framework of the EU Erasmus Mundus Action 2.

This framework is well adapted to biological systems but supposes a strong effort in the enumeration of all entities (and interactions between them) that play a role in the system. The third group of approaches consists of qualitative frameworks in which details have been abstracted and only main causalities have been taken into account. Two paradigmatic frameworks can be classified in this group: the Boolean networks first introduced by Kauffman [5] and the multi-valuated modelling framework first introduced by R. Thomas [13].

In this paper we present a modelling framework that links the well known modelling framework of gene network introduced by R. Thomas and Markov chains. In a first development we introduce a Markov chain having as state space the set of all possible states of the R. Thomas models: we generate the transition probabilities by examining all the possible parameterizations of the interaction graph. Thus the Markov chain represents the possible behaviours obtained by superposition of all parameterized models. We then extend this stochastic framework to a Markov chain in which we distinguish the states of each parameterized model and where the probabilities are computed on a smaller set of parameterizations. The idea is to consider only parameterized qualitative models of R. Thomas which represent well the biological knowledge and to use transition probabilities to allow the system to jump from a particular dynamics to another one.

The earliest qualitative model for a gene regulatory network was introduced by Kauffman [5]. In Kauffman's model, a gene is modelled as a binary variable (0 or 1) which takes only one of the possible Boolean functions of its inputs. When the gene is on it takes the value of 1, otherwise it takes 0. The outputs of a gene at time $t + 1$ depends only on the activity at time t. In this group of qualitative modelling frameworks, we can also cite the framework of R. Thomas in which each gene can have several levels of expression [13]. Thomas' model allows the gene to be represented as a multilevel logical variable $(0, 1, 2, \ldots)$; the number of possible values depends on the number of distinct actions it does on the network. In this case, the actions refer to a gene acting as an activator or repressor of some of the genes in the network. For each distinct action, a threshold value is assigned to specify from which expression level the influence takes place. So a variable with n distinct actions has n thresholds and this variable becomes an $(n+1)$-level variable. Allowing multilevel logical variables guarantees that no two distinct actions can happen simultaneously.

In order to illustrate our modelling approach, we focus on the gene regulatory network of the pathogen *Pseudomonas aeruginosa*, more specifically on the subsystem which is responsible for mucus production in the lungs of individuals with cystic fibrosis. Although the global gene regulatory network of

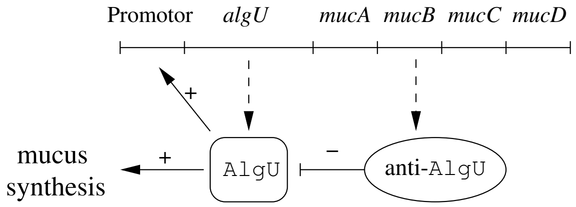


Figure 1: Portion of the gene regulatory network of the pathogen *Pseudomonas aeruginosa*, responsible for the mucus production; an arrow indicates activation or stimulation while a T-symbol represents repression [1, 2].

this pathogen consists of 690 genes and 1020 regulatory interactions between their products [3], the subsystem controlling the mucus production consists of some genes and proteins, see Figure 1. Because the mucus production worsens the respiratory problem of the patients which is often the cause of death [2, p. 75], elucidating the behaviour of this subsystem may be of great help to address this outcome.

The paper is organised as follows. Section 2 is devoted to sketch the qualitative modeling framework of R. Thomas. Section 3 explains how to build a Markov chain from the set of all possible parameterizations of an interaction graph. We can then push this idea forward and propose, when biological knowledge allows to reduce the set of possible parameterizations, a unique stochastic model where it becomes possible to jump from one qualitative model to another, see Section 4. Finally Section 5 is devoted to conclusion and discussion.

2 Reminding of R. Thomas' Modelling Framework

The biological regulatory network controlling the mucus production in *Pseudomonas aeruginosa* can be abstracted by the simple directed graphs of Figure 2 in which positive and negative signs indicate activation and repression, respectively, following the direction of the edge they label. These interaction graphs would suffice if we are only interested in applying Kauffman's model but if we want to apply Thomas' model there must be a threshold indicator for each distinct action of the gene as seen in Figure 2.

Such interaction graphs, as those in Figure 2, are called *biological regulatory graphs* [1, Definition 1] and are represented by graphs $G = (V, E)$, where V is the set of genes in the network and E represents the set of interactions between the genes in V. Each vertex $v \in V$ has a boundary b_v that is less than or equal to its out-degree (unless its out-degree is zero in which case we take the boundary to be one) while each edge is labelled by an ordered pair containing the threshold t and action ε (activation "+" or repression "−").

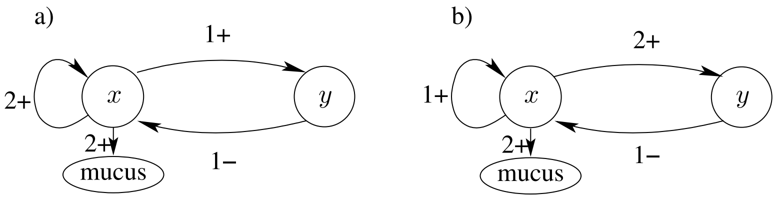


Figure 2: Two simple interaction graphs representing the system controlling the production of mucus in *Pseudomonas aeruginosa* (see Figure 1). The variable x denotes the gene algU and the protein AlgU while y denotes the gene mucB and the anti-AlgU. Both biological regulatory graphs differ by the labelling of edges outgoing from node x: the thresholds are not the same.

In Figure 2, we have $V = \{x, y\}$ and $E = \{(x \to x), (x \to y), (y \to x)\}$. The interaction $x \to$ mucus is not taken into consideration since mucus has no action backward toward x and y: it is a by-product of x which is produced when x is at its highest level (the regulation ($x \to$ mucus) is labelled by the threshold 2 since x has two distinct actions on y and on itself). The variable y has a unique action (repression of x) so the only possible threshold of the regulation ($y \to x$) is 1. Lastly note that Figure 2 does consider two possible biological regulatory graphs because the ordering between thresholds labelling edges outgoing from node x is not well known: we have to consider the two possible orderings.

In order to build the the dynamics of a biological regulatory graph, we first introduce the states of the network. A *state* of a regulatory network is a tuple denoted by $(n_{v_1}, \ldots, n_{v_p})$, where p denotes the number of genes and for each $n_{v_i} \in \mathbb{N}$ (natural numbers / non negative integers) $n_{v_i} \leq b_v$ [1, Definition 3]. We have now to define the *resources* of a vertex v_i with respect to a state $(n_{v_1}, \ldots, n_{v_p})$. Given a regulatory network, a state $(n_{v_1}, \ldots, n_{v_p})$ and an edge $(v_i \to v_j)$ with label (t, ε), the vertex v_i is a resource of v_j if and only if $n_{v_i} \geq t$ and $\varepsilon = +$ or $n_{v_i} < t$ and $\varepsilon = -$ [1, Definition 4]. The intuition is that the absence of an inhibitor plays the same role as the presence of an activator. Finally, a *biological regulatory network* refers to the biological regulatory graph $G = (V, E)$ together with a set of parameters $\mathscr{K} = \{k_{v,\omega}\}$, where $v \in V, \omega \subset G^{-1}(v) = \{u \mid (u \to v)$ is an edge in $G\}$ and $k_{v,\omega} \leq b_v$ [1, Definition 2]. The parameter $k_{v,\omega}$ gives the value towards which v is attracted when the set of resources of v is ω.

An easy way to represent the dynamics of a regulatory network is to associate with each state, the state towards which the system is attracted, when considering that each variable v changes at the same time to its current attraction value $k_{v,\omega}$ (ω being the current set of ressources of v). This defines the so-called *synchronous state graph* $\mathscr{S} = (S, T)$: The set of vertices S contains all possible states, and the edges of T are of the form $(n_{v_1}, \ldots, n_{v_p}) \rightarrow (k_{(v_1,\omega_1)}, \ldots, k_{(v_p,\omega_p)})$ such that for every i, ω_i is the set of resources of v_i at the state $(n_{v_1}, \ldots, n_{v_p})$ [1, Definition 5]. Unfortunately, the parameters $k_{v,\omega}$ are not measurable in vivo [1, p. 342] so we are left with several possibilities which results to obtaining several synchronous state graphs.

The synchronous state graph is not well adapted to represent evolution of the biological system because it is improbable that two (or more) genes reach their thresholds exactly at the same time and because a gene cannot directly jump two or more consecutive thresholds. To correct these drawbacks, one has to *desynchronize* each transition. Each transition $(n_{v_1}, \ldots, n_{v_p}) \rightarrow (n'_{v_1}, \ldots, n'_{v_p})$ is replaced by the set of its *desynchronizations* which are of the form $(n_{v_1}, \ldots, n_{v_i-1}, n_{v_i}, n_{v_i+1}, \ldots, n_{v_p}) \rightarrow (n_{v_1}, \ldots, n_{v_i-1}, n_{v_i} + \delta, n_{v_i+1}, \ldots, n_{v_p})$ for i such that $n_{v_i} \neq n'_{v_i}$ and $\delta = 1$ when $n_{v_i} < n'_{v_i}$, otherwise $\delta = -1$ [1, Definition 6]. The desynchronization step allows some states to transition to more than one other state. Thus, the dynamics of the regulatory graph is represented by the *asynchronous state graph* $\mathscr{S}' = (S, T')$ where the set S of vertices is the set of states and the set T' of transitions contains all desynchronized transitions of the synchronous state graph [1, Definition 7]. Note that two different synchronous state graphs may lead to the same asynchronous state graph since the desynchronization step can reduce two distinct synchronous transitions to the same set of desynchronized transitions.

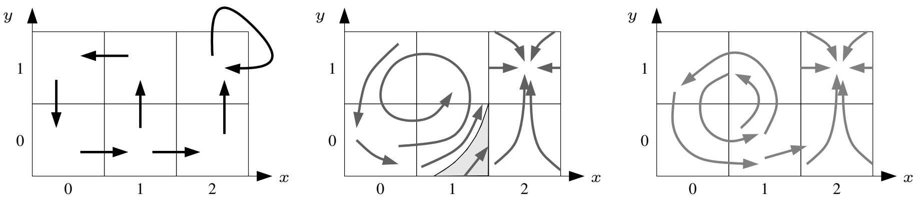


Figure 3: A possible qualitative dynamics in the modelling framework of R. Thomas (left). This asynchronous state graph can correspond to an inward spiral (centre) or to an outward spiral (right).

The global modelling approach consists of identifying all variables of the system, as well as their interactions and then the identification of parameters. Unfortunately, sometimes, it is not clear which parameters to choose. Consider the possible qualitative dynamics shown in Figure 3 (left) where we see that the state $(2, 1)$ is a stable state and that the system presents a counterclockwise oscillation between states which have a level of x less or equal to 1. It is clear that both the inward spiral (Figure 3 centre) and the outward spiral (right) are represented by the same qualitative model. But it could be more convenient to represent the inward spiral by the model where transition $(1, 0) \to (2, 0)$ does not appear: the small part of the domain $(1, 0)$ from which the stable domain $(2, 1)$ is reachable, is integrated in the domain $(2, 0)$, and when a grand tour is done in the inward spiral, it become impossible to reach the stable domain $(2, 1)$.

3 Markov Chains in Gene Regulatory Networks

When Kauffman proposed the Boolean model, the output of the genes at time $t + 1$ were only dependent on the activity of its inputs at time t [5, p. 441] which resembles the Markov property where given the past and the present, the future only depends on the present [9, p. 163]. In Kauffman's model a gene at time t transitions only to exactly one state at $t + 1$ while in a Markov chain, the transition probabilities allows the system to go from a state at time t to more than one other state at $t + 1$. In this section, we discuss briefly several ways of setting up Markov chains for gene regulatory networks based on available literature (see the works of Skornyakov *et al.* [12], Kim *et al.* [7] and Shmulevich *et al.* [11]) and then we give a basic Markov chain that represents the asynchronous dynamics of the interaction graphs of Figure 2.

In these three articles [7, 11, 12], a state can be thought of as a snapshot of the activity level of all the genes with respect to a given time. In [11], these states were referred to as maps. The Markov chain was applied to Kauffman's Boolean model of a gene regulatory network but it requires that the cooperation between interactions are well specified. In [7], the Markov chain allowed each gene to take three states, namely -1 (under-expressed), 0 (equivalently-expressed) and 1 (over-expressed) and it makes use of conditional probabilities to compute the transition probabilities. The Probabilistic Boolean Network (PBN) [11] addresses the deterministic nature of Kauffman's Boolean model. Both works [7, 11] can be easily extended to Thomas' model. However, the updates on all the genes in a PBN are done synchronously to simplify computation while preserving the generic properties of global network dynamics [11].

In Thomas' modelling framework, there are several possible values for the parameters k_{v_i,ω_i}, where v_i denotes the ith gene while ω_i denotes the ith gene's resources. The variability of these parameters results to potentially enormous (exponential) number of synchronous state graphs, but this number can be largely trimmed by considering the following constraints:

$$k_{v,\emptyset} = 0 \text{ and } \omega \subseteq \omega' \Rightarrow k_{v,\omega} \leq k_{v,\omega'}. \tag{1}$$

When a gene v has no resources, its expression level is not supposed to increase. Hence, a value of zero is assigned to $k_{v,\emptyset}$. When a gene loses some of its resources its expression level may drop while increasing its resources may increase its expression level. Because of the constraints in (1), the number of synchronous state graphs for each regulatory graph in Figure 2 is reduced to 28 which is now a reasonable number of graphs to work with. These synchronous state graphs can be obtained by playing with the different values of the parameters of Table 1 with the previous constraints of inclusion of resources in mind.

a)

State		Next State	
x	y	x	y
0	0	$k_{x,\{y\}}$	0
0	1	0	0
1	0	$k_{x,\{y\}}$	$k_{y,\{x\}}$
1	1	0	$k_{y,\{x\}}$
2	0	$k_{x,\{x,y\}}$	$k_{y,\{x\}}$
2	1	$k_{x,\{x\}}$	$k_{y,\{x\}}$

b)

State		Next State	
x	y	x	y
0	0	$k_{x,\{y\}}$	0
0	1	0	0
1	0	$k_{x,\{x,y\}}$	0
1	1	$k_{x,\{x\}}$	0
2	0	$k_{x,\{x,y\}}$	$k_{y,\{x\}}$
2	1	$k_{x,\{x\}}$	$k_{y,\{x\}}$

Table 1: Tables giving the parameters of interaction graphs according to the current state: Tables a) and b) correspond to Figures 2(a) and 2(b), respectively.

We now recall the foundations of Markov chains in discrete time on a countable state space. Let p_{ij}^n denote the probability that state j can be reached from state i in n steps. If $n = 1$ we have the entries p_{ij} of the transition matrix $\mathbf{P}$ while the n-step transition probabilities p_{ij}^n $(n > 1)$ are contained in the matrix $\mathbf{P}^n$. If for any couple of states (i, j) we can find an $n \in \mathbb{N}^+$ (positive integers) such that $p_{ij}^n > 0$ and $p_{ji}^n > 0$, then we say that the states *communicate* with each other. This indicates that all the states belong to a unique class (Markov chain is irreducible). A state i has period d if $p_{ii}^n = 0$ whenever n is not divisible by d and d is the greatest integer with this property [9, p. 169]. In an irreducible aperiodic Markov chain the states are either all transient or null recurrent (finite number of visits) or positive recurrent (infinite number of visits) with a unique stationary distribution $\{\pi_j, j = 1, 2, \ldots\}$, where $\pi_j = \lim_{n\to\infty} p_{ij}^n > 0$ [9, Theorem 4.3.3]. Note that an irreducible Markov chain with a finite state space cannot have transient states because the chain will eventually stop once it has visited all the states in a finite number of time which should not be the case [9, p. 170]. Thus, in such a chain, all the states must be positive recurrent.

Let μ_{jj} denote the expected number of transitions needed to return to state j starting from j. When state j is positive recurrent $\mu_{jj} < \infty$ [9, p. 173] and when the Markov chain is aperiodic and irreducible, we have $\lim_{n\to\infty} p_{ij}^n = 1/\mu_{jj}$ [9, Theorem 4.3.1]. It follows that in such a Markov chain, we have $\pi_j = 1/\mu_{jj}$. An aperiodic irreducible positive recurrent Markov chain is called *ergodic* [9, p. 177]. In an ergodic Markov chain we have a limiting matrix

$\mathbf{\Pi} = \lim_{n\to\infty} \mathbf{P}^n$ with all rows having the same vector $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots)$ of positive probabilities with sum equal to 1; the probability π_i denotes the long-run proportion of time that the Markov chain stays in state i

(A)	(0,0)	(0,1)	(1,0)	(1,1)	(2,0)	(2,1)
(0,0)	12	0	16	0	0	0
(0,1)	28	0	0	0	0	0
(1,0)	9	0	5	9.5	4.5	0
(1,1)	0	21	7	0	0	0
(2,0)	0	0	7.5	0	9	11.5
(2,1)	0	0	0	16.5	8.5	3

(B)	(0,0)	(0,1)	(1,0)	(1,1)	(2,0)	(2,1)
(0,0)	12	0	16	0	0	0
(0,1)	28	0	0	0	0	0
(1,0)	2	0	8	0	18	0
(1,1)	0	6	19	0	0	3
(2,0)	0	0	7.5	0	9	11.5
(2,1)	0	0	0	16.5	8.5	3

Table 2: Sum of the probabilities assigned to each possible transition over all asynchronous state graphs. (A) and (B) are built from Tables 1(a) and 1(b) respectively. These numbers take into account the multiplicity of asynchronous state graphs.

To model the asynchronous dynamics by a Markov chain, we examine all the possible asynchronous state graphs. Since different synchronous state graphs may lead to an identical asynchronous state graph, the number of distinct asynchronous state graphs can be less than the number of distinct synchronous state graphs. In that case, we also have to take into account the multiplicity (number of occurrences) of each distinct asynchronous state graph. In each asynchronous state graph, we assign appropriate probabilities to transitions (the transitions outgoing from a same state receive the same probability if no knowledge contradicts this hypothesis). Once this is done for each asynchronous state graph, we multiply the probabilities assigned to each possible transition by the multiplicity of the asynchronous state graph and take the sum of all such terms over all the possible asynchronous state graphs.

We now set-up the transition probability matrices for Tables 1(a) and 1(b) which are obtained by simply dividing the entries of Table 2 by the total number of synchronous state graphs which is 28 as already mentioned. We have:

$$\mathbf{P}_a = \begin{bmatrix} \frac{3}{7} & 0 & \frac{4}{7} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{9}{28} & 0 & \frac{5}{28} & \frac{19}{56} & \frac{9}{56} & 0 \\ 0 & \frac{3}{4} & \frac{1}{4} & 0 & 0 & 0 \\ 0 & 0 & \frac{15}{56} & 0 & \frac{9}{28} & \frac{23}{56} \\ 0 & 0 & 0 & \frac{33}{56} & \frac{17}{56} & \frac{3}{28} \end{bmatrix} \text{ and } \mathbf{P}_b = \begin{bmatrix} \frac{3}{7} & 0 & \frac{4}{7} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{14} & 0 & \frac{2}{7} & 0 & \frac{9}{14} & 0 \\ 0 & \frac{3}{14} & \frac{19}{28} & 0 & 0 & \frac{3}{28} \\ 0 & 0 & \frac{15}{56} & 0 & \frac{9}{28} & \frac{23}{56} \\ 0 & 0 & 0 & \frac{33}{56} & \frac{17}{56} & \frac{3}{28} \end{bmatrix}.$$

The order of the entries in $\mathbf{P}_a$ and $\mathbf{P}_b$ follow the order given in Table 2. These Markov chains are ergodic, which guarantees the existence of a unique stationary probability π_i which gives the long-run proportion of time in i.

- Evaluating $\lim_{n\to\infty}\mathbf{P}_a^n$, we have $\pi_{(0,0)} = 0.339$, $\pi_{(0,1)} = 0.096$, $\pi_{(1,0)} = 0.304$, $\pi_{(1,1)} = 0.128$, $\pi_{(2,0)} = 0.091$, and $\pi_{(2,1)} = 0.042$. In the long-run, the most visited states are (0,0), (1,0), and (1,1). Given the long-run probabilities, we can also compute the average number of transitions required to return to each state. Recall that μ_{ii} denotes the expected number of transitions needed to return to state i starting at state i. For this chain, the mean return times for the states $(0,0)$, $(0,1)$, $(1,0)$, $(1,1)$, $(2,0)$ and $(2,1)$ are given by $\mu_{11} = 2.949$, $\mu_{22} = 10.424$, $\mu_{33} = 3.285$, $\mu_{44} = 7.818$, $\mu_{55} = 11.014$, and $\mu_{66} = 23.944$, respectively.
- For $\mathbf{P}_b$, we obtain $\lim_{n\to\infty}\mathbf{P}_b^n$ which gives the stationary distributions $\pi_{(0,0)} = 0.073$, $\pi_{(0,1)} = 0.022$, $\pi_{(1,0)} = 0.285$, $\pi_{(1,1)} = 0.101$, $\pi_{(2,0)} = 0.347$, and $\pi_{(2,1)} = 0.172$. The most visited states are $(2,0)$, $(1,0)$, and $(2,1)$. For this chain, the mean return times are given by $\mu_{11} = 13.592$, $\mu_{22} = 46.125$, $\mu_{33} = 3.508$, $\mu_{44} = 9.884$, $\mu_{55} = 2.883$, and $\mu_{66} = 5.824$.

Both chains show that 70% of the time in the long-run $y = 0$ which means that x is not inhibited; we then expect that in the long-run $x \neq 0$ most of the time. On one hand, this is true for $\mathbf{P}_b$ since the most visited states in the long-run are the states (2, 0) and (1, 0). On the other hand, in the case of $\mathbf{P}_a$, these both states are visited only close to 40% of the time. Moreover a drawback of setting up a Markov chain this way is the inability to show the steady-states or circuits of the asynchronous state graphs.

4 Probabilistic Gene Network (PGN)

In the previous model, the Markov chain comes from a superposition of all the parameterized qualitative models. In order to distinguish the different asynchronous state graphs, we introduce a Markov chain which memorizes the asynchronous state graph a particular state is in. Because all asynchronous state graphs can differ drastically, we limit this Markov chain to a set of asynchronous state graphs that behave closely (see below).

In PGNs, we take into consideration attractors. The *attractors* of a network are the smallest sets of states from which one cannot escape [10, Section 2.5]. This can be a stable state which is a state without successors or a group of states that demonstrates sustained oscillations without exits. These latter attractors are said to be cyclic and, naturally, it is not possible to reach a stable state starting from a cyclic attractor. Note that every asynchronous state graph has at least an attractor. The stable states of an asynchronous state graph results to having absorbing states in the Markov chain built on it. The presence of absorbing states may result to obtaining an *absorbing Markov chain*; this happens when it is possible to eventually reach an absorbing state from every state [4, p. 416]. To show a Markov chain is absorbing, we need to find an $n \in \mathbb{N}^+$ such that all the entries of $\mathbf{P}^n$ are non zero, $\mathbf{P}$ being the transition probability matrix. An absorbing Markov chain give the expected times of absorption and the probability of absorption from every transient state.

Let $\mathscr{N} = \{N_1, \ldots\}$ denote a subset of Thomas' networks (asynchronous state graphs). This set can correspond to all asynchronous state graphs that are coherent with some biological knowledge, in that sense, the set is supposed to be largely smaller than the total number of asynchronous state graphs. This set can result e.g. from a filtering step which selects only asynchronous state graphs which are coherent with behavioural properties expressed in a formal language [1]. We introduce an ordering relation between Thomas' networks: for distinct networks N and N', we have $N > N'$ if and only if N' is a subgraph of N. This ordering relation leads to consider the set of models equipped with this relation as a lattice with possibly several minimal elements. Denote by $S = \{s_1, s_2, \ldots\}$ a set of states (this set is common to all asynchronous state graphs).

The intuition is the following. A biological system can be represented by a set of different dynamics (asynchronous state graphs). In a particular environment, the biological system can behave exactly as one of these dynamics but according to some changes of the environment, the behaviour of the biological system can adopt the dynamics of another asynchronous state graph. It becomes natural to allow the Markov chain to jump from one asynchronous state graph to another. But it is unlikely that the biological system jumps from a state of a certain network toward another state in another network with a very different dynamics from the initial network. This is the reason why the jumps are possible only under some conditions on the ordering relation between Thomas' networks.

A probabilistic gene network (PGN for short) on $\mathscr{N}$ satisfies the following conditions:

i. For each $N \in \mathscr{N}$ and each transition $s \to s'$ on the asynchronous state graph N a probability of $P_N(s \to s')$ is attached in such a way that for every s, $\sum_{s' \in S} P_N(s \to s') = 1$.

ii. For each pair of networks $N, N' \in \mathscr{N}$ such that $N > N'$ and there exists no other network N'' such that $N > N'' > N'$, a probability is attached to $(N \to N')$ in such a way that the sum of all such probability for a given N is less than 1.

Once a probabilistic gene network has been established, we define its corresponding probabilistic state graph. Let (N, s) denote the set of nodes of the probabilistic state graph, where $N \in \mathscr{N}$ and $s \in S$. The set of edges (transitions) is defined by $(N, s) \to (N', s')$ if and only if $s \to s'$ (with $s' \neq s$) is a transition of N and $P(N \to N') \neq 0$. The probability $P((N, s) \to (N', s'))$ is defined by

$$\begin{aligned} &P((N,s) \to (N',s')) \\ &\quad = \frac{W(N')\mathbf{1}_{[(s\to s')\in N]}(N)P_{N'}(s \to s')}{\displaystyle\sum_{N^* \in \mathscr{N}} \sum_{s^* \in S} W(N^*)\mathbf{1}_{[(s\to s^*)\in N^*]}(N^*)P_{N^*}(s \to s^*)}, \end{aligned} \tag{2}$$

where $\mathbf{1}_A(\cdot)$ is the indicator function and $W(N)$ denotes the weight of the network N which pertains to its multiplicity in the set of all possible asynchronous state graphs of $\mathscr{N}$. When the transition does not involve a change of network, we replace N' by N to compute for the corresponding probability.

The probabilistic state graph is used to set up a Markov chain that would hopefully give a better representation of the dynamics of a gene regulatory network. More precisely the probabilistic state graph allows one to walk inside the set of considered asynchronous state graphs. Thus, a probabilistic gene network gives information not only on the state of each genes but also the network containing the state.

We present an illustration of a Probabilistic Gene Network by considering the networks of Figure 4. For $N = N_1$ (resp. N_2), the value of $P_N(s \to s')$ is computed making the assumption that in the network N the probability of transitioning from s to any of its successor is equally likely which is 1 divided by the number of transitions outgoing from s in N. Moreover we suppose that $\mathscr{N}$ contains twice N_1 and once N_2, which can be interpreted as: staying in N_1 is more likely than jumping from N_1 to N_2. The transition probability matrix is:

$$\begin{array}{r} N_1,(0,0) \\ N_1,(0,1) \\ N_1,(1,0) \\ N_1,(1,1) \\ N_1,(2,0) \\ N_1,(2,1) \\ N_2,(0,0) \\ N_2,(0,1) \\ N_2,(1,0) \\ N_2,(1,1) \\ N_2,(2,0) \\ N_2,(2,1) \end{array} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2/3 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0 \\ 2/3 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 1/6 & 1/6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

We can note that it is not possible to escape from $(N_1, (0, 0))$ and $(N_1, (2, 0))$. This is due to the fact that to jump from an asynchronous state graph to another one, it is mandatory to be in a state which is not stable in the initial asynchronous state graph. Thus stable states in non minimal asynchronous state graphs (in the lattice) become absorbing states of the chain. In this illustration, the resulting Markov chain is an absorbing Markov chain so we can generate the expected time of absorption from each transient state and its corresponding probability of absorption in the stable states as shown in Tables 3 and 4, respectively.

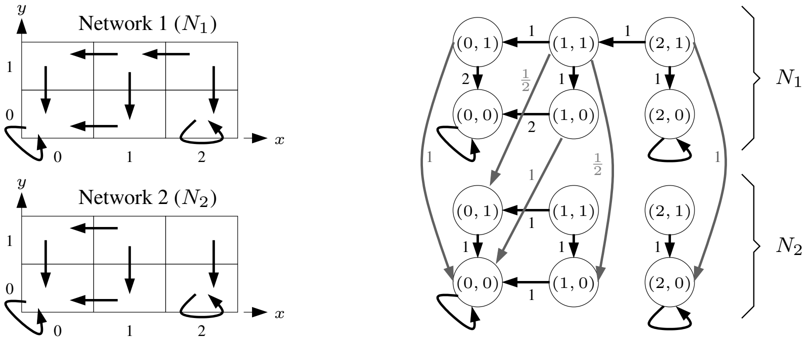


Figure 4: Two asynchronous state graphs (left) associated with interaction graph of Figure 2(a) on which we illustrate the construction of the probabilistic gene network (right). The numbers labelling the transitions (right) correspond to the numerator of Equation 2.

When involved networks have only cyclic attractors, the resulting Markov chain gets divided into several classes whose number would depend on the number of networks in the subset and on the structure of the lattice. Because the chain only allows a transition from a larger network to a smaller one, all the states in a network form a class of transient states except the states in the smallest networks (according to the lattice structure of the set of networks) which constitute different classes of recurrent states.

Transient States	Expected Time of Absorption
$(N_1,(0,1))$	1
$(N_1,(1,0))$	1
$(N_1,(1,1))$	2
$(N_1,(2,1))$	5/3 = 1.666

Transient States	Expected Time of Absorption
$(N_2,(0,1))$	1
$(N_2,(1,0))$	1
$(N_2,(1,1))$	2
$(N_2,(2,1))$	1

Table 3: Expected time of absorption (number of transitions) from any of transient state to any of the absorbing states for the networks in Figure 4.

Transient	Probability of Absorption			
States	$(N_1,(0,0))$	$(N_1,(2,0))$	$(N_2,(0,0))$	$(N_2,(2,0))$
$(N_1,(0,1))$	2/3	0	1/3	0
$(N_1,(1,0))$	2/3	0	1/3	0
$(N_1,(1,1))$	4/9	0	5/9	0
$(N_1,(2,1))$	4/27	1/3	5/27	1/3
$(N_2,(0,1))$	0	0	1	0
$(N_2,(1,0))$	0	0	1	0
$(N_2,(1,1))$	0	0	1	0
$(N_2,(2,1))$	0	0	0	1

Table 4: The transient states in the smaller network can only be absorbed in its own steady states because it is not allowed to leave the network. The transient states in the bigger network can be absorbed by any of the absorbing states of the networks under consideration, see Figure 4.

5 Discussion and conclusion

In this article we mixed two different frameworks of gene networks allowing to take advantage of the formal framework of R. Thomas modelling theory and to use transition probabilities of Markov chains to change the parameterized model. According to Figure 3, the modelling framework of R. Thomas leads to a single unique model, both the inward spiral and the outward spiral. In a natural way, it could be more efficient to represent the inward spiral by the model where transition $(1,0) \rightarrow (2,0)$ does not appear. In such a case, it could be interesting to consider in a unique framework both discrete state graphs and to allow the trajectory to jump from one to another, if information coming from biological experiments reinforces the belief in a particular model. For example, longer are the observed traces around the qualitative cycle, bigger the belief in the model representing the inward spiral.

In such a way, it becomes natural to consider each asynchronous state graph as the dynamics of the biological system in a particular context. When the environment changes the context, the qualitative dynamics can also change. Probabilistic Gene Networks presented in this article, constitute a first framework allowing to jump from a qualitative dynamics to another one.

References

[1] Bernot, G., J-P. Comet, A. Richard & J. Guespin, (2004) Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic. *Journal of Theoretical Biology* **229**: 339-347.

[2] Bernot, G., J-P. Comet, A. Richard, M. Chaves, J-L. Gouzé, J-L. & F. Dayan, (2013) Modeling and analysis of gene regulatory networks. In *Modeling in Computational Biology and Biomedicine: A Multidisciplinary Endeavor*, eds F. Cazals, P. Kornprobst, pp. 47-80. Springer-Verlag Berlin Heidelberg.

[3] Galán-Vásquez, E., B. Luna, & A. Martinez-Antonio, (2011) The regulatory network of *Pseudomonas aeruginosa*. *Microbial Informatics and Experimentation* **1(1)**, 3. doi:10.1186/2042-5783-1-3

[4] Grinstead, C.M. & J.L. Snell, (1997) Introduction to Probability, 2nd edn (pp. 415-422). *American Mathematical Society*.

[5] Kauffman, S.A., (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology* **22**: 437-467.

[6] Klipp, E., R. Herwig, A. Kowad, C. Wierling & H. Lehrach, (2005) Systems Biology in Practice: Concepts, Implementation and Application. *Wiley-VCH*, Weinheim.

[7] Kim, S., H. Li, H., E.R. Dougherty, N. Cao, Y. Chen, M. Bittner, & E.B. Suh, (2002) Can Markov chain models mimic biological regulations? *Journal of Biological Systems* **10(4)**: 337-357.

[8] National Institutes of Health. *Talking Glossary of Genetic Terms*. National Human Genome Research Institute.

[9] Ross, S.M., (1996) Stochastic Processes, 2nd edn. *John Wiley & Sons, Inc.*, USA.

[10] Richard, A., J-P. Comet & G. Bernot, (2008) R. Thomas' logical model. Available from: http://www.i3s.unice.fr/ bernot/Teaching/2008-LilleSchool-RichardCometBernot.pdf

[11] Shmulevich, I., E.R. Dougherty, S. Kim & W. Zhang, (2002) Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics* **18(2)**, 261-274.

[12] Skornyakov, V.P., M.V. Skoryakova, A.V. Shurygina & P.V. Skornyakov, (2014) *Finite-state discrete-time Markov chain models of gene regulatory networks*. [Preprint] Available from: http://biorxiv.org/

[13] Thomas, R., (1991). Regulatory networks seen as asynchronous automata: a logical description. *Journal of Theoretical Biology* **153**: 1-23.

EFM-Recorder – Faster Elementary Mode Enumeration via Reaction Coupling Order

Annika Röhl, Yaron Goldstein and Alexander Bockmayr

Department of Mathematics and Computer Science, Freie Universität Berlin, Arnimallee 6, 14195 Berlin, Germany, annika.roehl@fu-berlin.de

Abstract

In metabolic network analysis, *Elementary Flux Modes* (EFMs) play an important role. Although they are of great interest, it is still hard to compute them, due to their large number. Therefore, it is desirable to develop methods which simplify the computation of EFMs in a given network. Here, we present a new approach that uses flux coupling relations in order to reduce the search space for EFM computation. *Flux Coupling Analysis* (FCA) determines coupling relations between the reactions in a network. Two reactions are partially coupled, if zero flux through one reaction implies zero flux through the other. Whenever there are several reactions that are partially coupled to each other, it is sufficient to take one reaction of this class as a *representative*. There is no need to check if the other reactions in this class are carrying flux or not. Coupling relations are used to speed up the mixed-integer optimisation method for computing EFMs proposed by de Figueiredo et al. (2009). With the help of the representatives, the number of binary variables and consequently the size of the search space can be significantly reduced. Including additional information on directional coupling relations leads to further improvements.

1 Introduction

In systems biology, genome-scale metabolic network reconstructions are used to develop an *in silico* model of a system. The metabolic network is assumed to be in steady state, i.e., we consider the so-called flux cone $C = \left\{v \in \mathbb{R}^{\mathcal{R}} \mid Sv = 0,\ v_{\mathrm{Irrev}} \geq 0\right\}$ of all flux distributions over the network at steady state. Here, $S \in \mathbb{R}^{\mathcal{M}\times\mathcal{R}}$ denotes the stoichiometric matrix for a set of (internal) metabolites $\mathcal{M}$ and a set of reactions $\mathcal{R}$. The set $\mathrm{Irrev} \subseteq \mathcal{R}$ contains the irreversible reactions. The vectors $v \in C$ are called (feasible) flux vectors and can be interpreted as pathways in the corresponding metabolic network.

Definition 1 (Support)**.** *The* support *of a flux vector* $v \in \mathbb{R}^{\mathcal{R}}$ *is the set of active reactions in* v*:* $\mathrm{supp}(v) = \{i \in \mathcal{R} \mid v_i \neq 0\}$.

We can now define the elementary flux modes of a metabolic network [1].

Definition 2 (Elementary flux mode (EFM))**.** *An* elementary flux mode *(EFM) is a feasible flux vector $e \in C \setminus \{0\}$ which has minimal support with respect to set inclusion, i.e., there exists no feasible flux vector $v \in C \setminus \{0\}$ with* $\mathrm{supp}(v) \subsetneq \mathrm{supp}(e)$.

EFMs define minimal sets of reactions that can operate together in steady state. Minimal in the biological context means: If any of the reactions is deleted, then the whole flux cannot operate in steady state anymore. EFMs are a popular approach to analyse metabolic networks because every behaviour of the network can be represented with the help of the EFMs [1,2]. For a recent survey on EFMs and their applications, we refer to [3]. Although the EFMs are of great interest, it is still a demanding task to enumerate them. Different approaches haven been proposed, such as the *Double Description Method* [4] and refinements [5], or mixed-integer programming methods like the algorithm of de Figueiredo et al. [6].

One way to understand the topology and robustness of a metabolic network is *Flux Coupling Analysis* (FCA) [7]. It can be performed quite fast [8] and recently has been generalized via a lattice-theoretic framework to arbitrary qualitative models [9]. Here we will formally define a partial order induced by the coupling relation of reactions. We will show how this leads to a reduced search space for algorithms that use binary variables to indicate whether a reaction is used or not. This allows us to define an improved version of the algorithm in [6] so that we can enumerate elementary modes with smaller mixed-integer linear programs (MILPs). We implemented our resulting algorithm and discuss runtime advantages of our implementation `EFM-Recorder` (EFM enumeration via reaction coupling order) in comparison with previous approaches.

2 Reaction Coupling Order

As mentioned before we assume the network being in steady state. Thus we consider the steady state flux cone $C = \left\{v \in \mathbb{R}^{\mathcal{R}} \mid Sv = 0,\ v_{\mathrm{Irrev}} \geq 0\right\}$.

Definition 3 (Blocked reactions)**.** *A reaction $r \in \mathcal{R}$ is called* blocked *iff $v_r = 0$ for all $v \in C$.*

Since blocked reactions can never occur in an EFM, we assume from now on that they have been removed from the network.

For two unblocked reactions $r, s \in \mathcal{R}$ we can define three different coupling relationships [7,8].

Definition 4 (Coupling relations)**.**

$r \overset{=0}{\rightarrow} s$**:** *$s$ is* directionally coupled *to r iff $v_r = 0$ implies $v_s = 0$ for all $v \in C$.*

$r \overset{=0}{\leftrightarrow} s$**:** *$s$ is* partially coupled *to r iff $v_r = 0 \;\Leftrightarrow v_s = 0$ for all $v \in C$.*

$r \sim s$**:** *s is* fully coupled *to r iff there exist $\lambda \neq 0$ such that $v_r = \lambda v_s$ for all $v \in C$.*

Remark 2.1. *If two reactions are fully coupled, then they are also partially coupled (but not necessarily the converse).*

The coupling relation $\overset{=0}{\rightarrow}$ is reflexive and transitive, and thus defines a *preorder* [10]. The relation $\overset{=0}{\leftrightarrow}$ is also symmetric and therefore an *equivalence relation*. This means that the set of reactions $\mathcal{R}$ can be partitioned into equivalence classes $[r] = [r]_{\overset{=0}{\leftrightarrow}} = \{s \in \mathcal{R} \mid r \overset{=0}{\leftrightarrow} s\}$. We have $\mathcal{R} = \bigcup_{[r] \in \overline{\mathcal{R}}} [r]$, where $\overline{\mathcal{R}} = \mathcal{R}/\overset{=0}{\leftrightarrow}$ denotes the set of all equivalence classes.

An equivalence class can be represented by any of its elements. We say that r is a *representative* of $[r]$ or that $[r]$ is the *coupling class* of r. Note that $[r] = [s]$ iff $r \overset{=0}{\leftrightarrow} s$. Coupling classes are similar to the *enzyme subsets* introduced by Pfeiffer et al. [11]. Enzyme subsets are groups of reactions that are fully coupled to each other. Here, we relax this condition and also consider reactions that are only partially coupled.

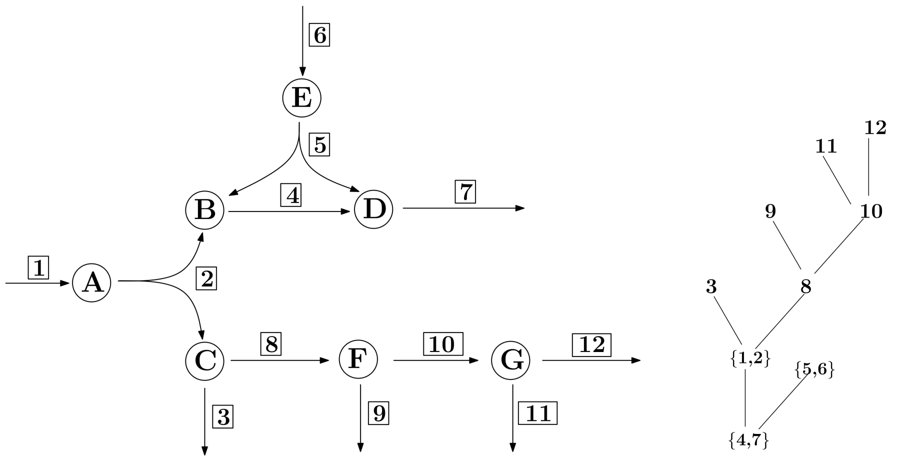
Definition 5 (Reaction coupling order $\prec_{\overset{=0}{\rightarrow}}$)**.** *The partial ordering on the coupling classes $\prec_{\overset{=0}{\rightarrow}} \subseteq \overline{\mathcal{R}} \times \overline{\mathcal{R}}$ defined by*

$$[r] \prec_{\overset{=0}{\rightarrow}} [s] :\Leftrightarrow r \overset{=0}{\rightarrow} s$$

is called the reaction coupling order (rcorder) *induced by the coupling relation* $\overset{=0}{\rightarrow}$.

Note that his construction works, because $\overset{=0}{\rightarrow}$ is a preorder [10].

Fig. 1 shows an example network with its corresponding rcorder. The network in Fig. 1a contains two pairs of fully coupled reactions, namely $\{r_1, r_2\}$ and $\{r_5, r_6\}$. The reactions $\{r_4, r_7\}$ are partially coupled. These three sets can be represented by r_1, r_5 and r_4 or just the indices $1, 5, 4$. Fig. 1b now shows the Hasse diagram of $\prec_{\overset{=0}{\rightarrow}}$, where the nodes represent reactions. If a reaction has zero flux, then exactly those reactions that are connected by a path going strictly upwards have zero flux, too. For example, reaction r_1 is



(a) Reactions $\{r_1, r_2\}$, $\{r_5, r_6\}$ and $\{r_4, r_7\}$ are coupling classes, where $\{r_4, r_7\}$ are partially but not fully coupled.

(b) Hasse diagram of rcorder $\prec_{\underset{\rightarrow}{=0}}$

Figure 1: Reaction coupling $\overset{=0}{\rightarrow}$ is a preorder of the reactions. It induces a partial order $\prec_{\underset{\rightarrow}{=0}}$.
A knock out of reaction r_1 implies inactivity in reaction r_3, r_8 and others, but not in reaction r_5. Thus, $1\overset{=0}{\rightarrow}3$ and $1\overset{=0}{\rightarrow}8$ but $\neg\left(1\overset{=0}{\rightarrow}5\right)$. This results in $1 \prec_{\underset{\rightarrow}{=0}} 3$ and $1 \prec_{\underset{\rightarrow}{=0}} 8$ for the rcorder $\prec_{\underset{\rightarrow}{=0}}$, but 1 and 5 are incomparable.
The Hasse diagram visualizes this by having downward directed paths starting in the greater and ending in the smaller of each pair of coupled reaction representatives, in our example from 3 to 1 and from 8 to 1, but not from 5 to 1. A knock out of reaction 4 implies inactivity in all reactions.

coupled to r_3, i.e., $r_1\overset{=0}{\rightarrow}r_3$, but r_1 is uncoupled to reaction r_5. More on Hasse diagrams can be found in [10].
One of the main advantages of coupling classes is that, if one reaction of a class is not carrying flux, no other reaction in the class does, and vice versa. Therefore, in every approach where binary variables are used to indicate if a reaction appears or not, it is enough to consider one reaction from every coupling class instead of considering all of them. Depending on the number of reactions and associated coupling classes, this may significantly reduce the number of variables that are needed.

3 Application to EFM computation

In the following we assume that the networks consist only of irreversible reactions. If the original network contains reversible reactions, then these reactions can be split into two irreversible reactions (one for every direction of the reversible one), see e.g. [12].

3.1 Finding EFMs with less binary variables

In 2009, de Figueiredo et al. [6] introduced an algorithm that allows enumerating elementary modes (EFMs) of the steady state flux cone using mixed-integer linear programming (MILP):

$$
\begin{aligned}
(\textbf{OriginalMILP})\ & \min \sum_{i\in\mathcal{R}} a_i && (1)\\
\text{s.t.}\quad & Sv = 0 && (2)\\
& a_i \le v_i \quad \forall i \in \mathcal{R} && (3)\\
& v_i \le M \cdot a_i \quad \forall i \in \mathcal{R} && (4)\\
& \sum_{i\in\mathcal{R}} a_i \ge 1 && (5)\\
& \sum_{i\in\mathcal{R}} Z_i^k a_i \le (\sum_{i\in\mathcal{R}} Z_i^k) - 1 \quad \forall k && (6)\\
& a_i \in \{0,1\}, v_i \ge 0.
\end{aligned}
$$

The algorithm minimizes (1) the number of active reactions in a steady state flux vector (2). As before, S denotes the stoichiometric matrix and v is a flux vector. To determine the active reactions, binary variables are used such that $a_i = 1$ iff reaction r_i is carrying flux. If $a_i = 0$, then $v_i = 0$, see (4), thus reaction r_i is not allowed to carry flux. Here, $M \gg 0$ is some big constant ("Big M"). Conversely, $a_i = 1$ implies $v_i \ge 1$, which is ensured by (3). To get a feasible flux vector different from the zero flux, (5) forces at least one reaction to be active. By definition, Z_i^k equals 1 if reaction r_i is carrying flux in the EFM which was computed in the k-th step, otherwise Z_i^k is 0. Thus (6) guarantees that the EFMs which were computed in the previous steps are not enumerated again. For more details, we refer to [6].

Based on the reaction coupling order, we can now use binary variables corresponding to the coupling classes $[r]$ instead of using binary variables for every individual reaction. Thus we can rewrite the algorithm of de Figueiredo et al. in the following way:

$$
\begin{aligned}
\min \quad & \sum_{[r]\in\overline{\mathcal{R}}} |[r]|\, a_{[r]} && (7)\\
\text{s.t.} \quad & Sv = 0 \\
& a_{[r]} \leq v_s \quad \forall [r] \in \overline{\mathcal{R}} \text{ and } \forall s \in [r] \\
& v_s \leq M \cdot a_{[r]} \quad \forall [r] \in \overline{\mathcal{R}} \text{ and } \forall s \in [r] \\
& \sum_{[r]\in\overline{\mathcal{R}}} a_{[r]} \geq 1 \quad \forall [r] \in \overline{\mathcal{R}} \\
& \sum_{[r]\in\overline{\mathcal{R}}} Z_i^k a_{[r]} \leq (\sum_{[r]\in\overline{\mathcal{R}}} Z_i^k) - 1 \quad \forall k \\
& a_{[r]} \in \{0, 1\} \\
& v_i \geq 0
\end{aligned}
$$

$|[r]|$ in (7) denotes the cardinality of the coupling class $[r]$. Thus, we compute the *shortest* EFMs w.r.t. the number of reactions and not the number of representatives.

The main advantage of our method is that we need only $\left|\overline{\mathcal{R}}\right|$ instead of $|\mathcal{R}|$ binary variables. For many genome-wide networks, this reduces the number of 0-1 variables by about 1/2, as shown in Tab. 2.

To further improve our approach, we may add the coupling constraints (8) and explicitly help the solver to set coupled variables to their correct values:

$$a_{[r]} \leq v_s \text{ if } s \overset{=0}{\rightarrow} t \text{ with } t \in [r] \tag{8}$$

Here we use directional coupling properties of the representatives. With this additional information, we do not reduce the number of binaries, but may speed up the running time of the algorithm.

3.2 Computational results

In a preprocessing step, we identified blocked and coupled reactions for different genome-wide network reconstructions using the software F2FC [8]. The results are given in Tab. 1. From this, we created Tab. 2, which shows the effect of using coupling classes instead of the original set of (unblocked) reactions. For most of the networks it is sufficient to work with as few as a third of the original number of reactions.

Next we ran our implementation EFM-Recorder on different metabolic networks for different choices of binary variables and constraints. All computations were done on a desktop machine with two processors Intel(R)

Model	**unblocked reactions**	**fully coupled**	**partially coupled**	**time in seconds**
Human recon 2	5837	5468	220	1801.1
E. coli iJO1366	2369	2143	2308	199.8
E. coli iAF1260	2167	1867	396	126.7
S. cerevisiae iND750	744	791	84	19.8
M. tuberculosis iNJ661	800	8567	7588	19.3
S. aureus iSB619	583	1204	874	11.7
H. pylori iIT341	501	4167	5212	6.4
E. coli textbook	95	44	0	0.48

Table 1: Number of reaction couplings (computed with `F2FC` [8]) for different genome-wide metabolic networks and the corresponding running times.

Model	**reactions**	**unblocked**	**representatives**
Human recon 2	7440	5837	4032
E. coli iJO1366	2583	2369	1399
E. coli iAF1260	2382	2167	1276
S. cerevisiae iND750	1266	744	446
M. tuberculosis iNJ661	1025	800	412
S. aureus iSB619	743	583	292
H. pylori iIT341	554	501	209
E. coli textbook	95	95	60

Table 2: Number of representatives for different genome-wide metabolic networks (computed with the `F2FC` [8]).

Model	EMs	Time ratio	
		`reps`	`coup`
E. coli textbook	10	8.7	27.7
	100	13.7	38.9
	1000	96.5	220.7
H. pylori iIT341	10	0.66	0.7
	100	23.7	32.6
	1000	187.1	167.7
S. aureus iSB619	10	4.2	5
	100	64.1	88.6
	1000	178	239.4
M. tuberculosis iNJ661	10	0.2	0.3
	100	1.4	2.9
	1000	0.4	1.1
S. cerevisiae iND750	10	4	4.4
	100	64.9	98.7
	1000		

Table 3: Speed up of the algorithms compared to the standard algorithm by de Figueiredo et al. [6]. For example, 8.6986 means that the method `reps` is 8.36986 times faster then `all`.

Core(TM) i5-2400S, CPU 2.50GHZ, each 2 threads. Tab. 4 shows how long it takes to calculate a desired number of EFMs, namely 10, 100 and 1000. In Tab. 3 the time ratio of the original algorithm compared to the ones introduced here are shown. Using coupling classes results in smaller MILPs (less binary variables), so we can expect shorter running times. The results in Tab. 4 meet these expectations especially for a large number of EFMs. In most cases, the method `coup` combining coupling classes with directional coupling constraints yields the best results.

Model	**Nr. of EMs**	**Method**		
		`all`	`reps`	`coup`
E. coli textbook	10	15.8	1.8	0.57
	100	323.9	23.6	8.3
	1000	20524	212.7	93
H. pylori iIT341	10	11	16.6	15.6
	100	1058.5	44.7	32.5
	1000	167490	894.9	999
S. aureus iSB619	10	29.2	7	5.9
	100	3281.7	51.2	37
	1000	107240	602.4	448
M. tuberculosis iNJ661	10	10.1	42.8	38.4
	100	199.5	138.6	68.6
	1000	2056	5182.4	1835.9
S. cerevisiae iND750	10	29.2	7.3	6.7
	100	6151.1	94.7	62.3
	1000			

Table 4: Time (in secs) needed to compute a given number of EFMs for different modelling approaches.
all: Each unblocked reaction r has its own binary variable $a_r = 1 \Leftrightarrow v_r \geq 1$.
reps: Only coupling class representatives $[r] \in \overline{\mathcal{R}}$ have binary variables $a_{[r]} = 1 \Leftrightarrow v_s \geq 1$ for $s \overset{=0}{\leftrightarrow} t$, with $t \in [r]$.
coup: Same as `reps`, but with additional directional coupling constraints $a_{[r]} \leq v_s$, for all $[r] \in \overline{\mathcal{R}}$ with $s \overset{=0}{\rightarrow} t$, where $t \in [r]$.

References

[1] S. Schuster and C. Hilgetag, "On elementary flux modes in biochemical reaction systems at steady state," *Journal of Biological Systems*, vol. 2, no. 02, pp. 165–182, 1994.

[2] S. Schuster, D. A. Fell, and T. Dandekar, "A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks," *Nature biotechnology*, vol. 18, no. 3, pp. 326–332, 2000.

[3] J. Zanghellini, D. E. Ruckerbauer, M. Hanscho, and C. Jungreuthmayer, "Elementary flux modes in a nutshell: Properties, calculation and applications," *Biotechnology journal*, vol. 8, no. 9, pp. 1009–1016, 2013.

[4] K. Fukuda and A. Prodon, "Double description method revisited," in *Combinatorics and computer science*, pp. 91–111, Springer, 1996.

[5] M. Terzer and J. Stelling, "Large-scale computation of elementary flux modes with bit pattern trees," *Bioinformatics*, vol. 24, no. 19, pp. 2229–2235, 2008.

[6] L. F. De Figueiredo, A. Podhorski, A. Rubio, C. Kaleta, J. E. Beasley, S. Schuster, and F. J. Planes, "Computing the shortest elementary flux modes in genome-scale metabolic networks," *Bioinformatics*, vol. 25, no. 23, pp. 3158–3165, 2009.

[7] A. P. Burgard, E. V. Nikolaev, C. H. Schilling, and C. D. Maranas, "Flux coupling analysis of genome-scale metabolic network reconstructions," *Genome research*, vol. 14, no. 2, pp. 301–312, 2004.

[8] A. Larhlimi, L. David, J. Selbig, and A. Bockmayr, "F2C2: a fast tool for the computation of flux coupling in genome-scale metabolic networks," *BMC bioinformatics*, vol. 13, no. 1, p. 57, 2012.

[9] Y. A. Goldstein and A. Bockmayr, "A lattice-theoretic framework for metabolic pathway analysis," in *Computational Methods in Systems Biology*, pp. 178–191, Springer, 2013.

[10] B. S. W. Schröder, *Ordered sets: an introduction*. Springer, 2003.

[11] T. Pfeiffer, J. Nu, F. Montero, S. Schuster, *et al.*, "Metatool: for studying metabolic networks.," *Bioinformatics*, vol. 15, no. 3, pp. 251–257, 1999.

[12] M. Terzer, *Large scale methods to enumerate extreme rays and elementary modes*. PhD thesis, Diss., Eidgenössische Technische Hochschule ETH Zürich, Nr. 18538, 2009, 2009.

Extensions for LTL model checking of Thomas networks

Adam Streck*[1], Heike Siebert [1]

[1] Freie Universität Berlin, Germany

Abstract

In the article we revisit the study [1], which described novel extensions to the method of model checking Thomas networks, which is a computer science approach to analysis of dynamical behaviour of biological systems. Here we provide new algorithms for the methods described in [1] and show how to extend these so that a wider range of biological data can be used.

1 Introduction

One of the main bottlenecks of modelling in systems biology is the absence of kinetic parameters and the complexity of identifying them. The scarcity of data and the non-linearity of the systems in question makes the task of reverse engineering in biology usually very difficult. Qualitative approaches try to circumvent the problem by using coarse abstractions under the assumption that biological systems are robust to changes in their environment. In turn, due to the simplicity of the abstract models, it is possible to completely describe and analyse their emergent behaviour. Here we focus on the modelling framework of R. Thomas [2] where behaviour of a system is described via multi-valued logical functions. If the kinetic parameters and consequently the logical functions are unknown, one can enumerate all the possibilities and select only those that fulfil certain constraints. This process is usually referred to as *parameter identification*. This is in our work conducted via *model checking* [3], which is one of the most prominent methods for automated model analysis.

In the article [1] we showed how to extend the model checking process to obtain additional knowledge. In particular we argued that when fitting a model to time-series data, the models that reproduce the data using with a small number of changes to the state of the system (e.g. producing and degrading a protein) are more preferable. We also provided paths of the system that match the behaviour and lastly assessed its robustness. However, the method of [1] was constrained to time series data, whereas here we extend it to the full expressiveness of the underlying model checking approach. Also, in [1] we were checking multiple models (usually 32) at once. While this approach provided a performance boost in practice, we found it too complex to describe

*corresponding author: adam.streck@fu-berlin.de

or extend and here we therefore only work with one model at a time. In summary, we present algorithms for the methods of [1], show that they have good time and space complexity, and provide functional extensions to them.

2 Background

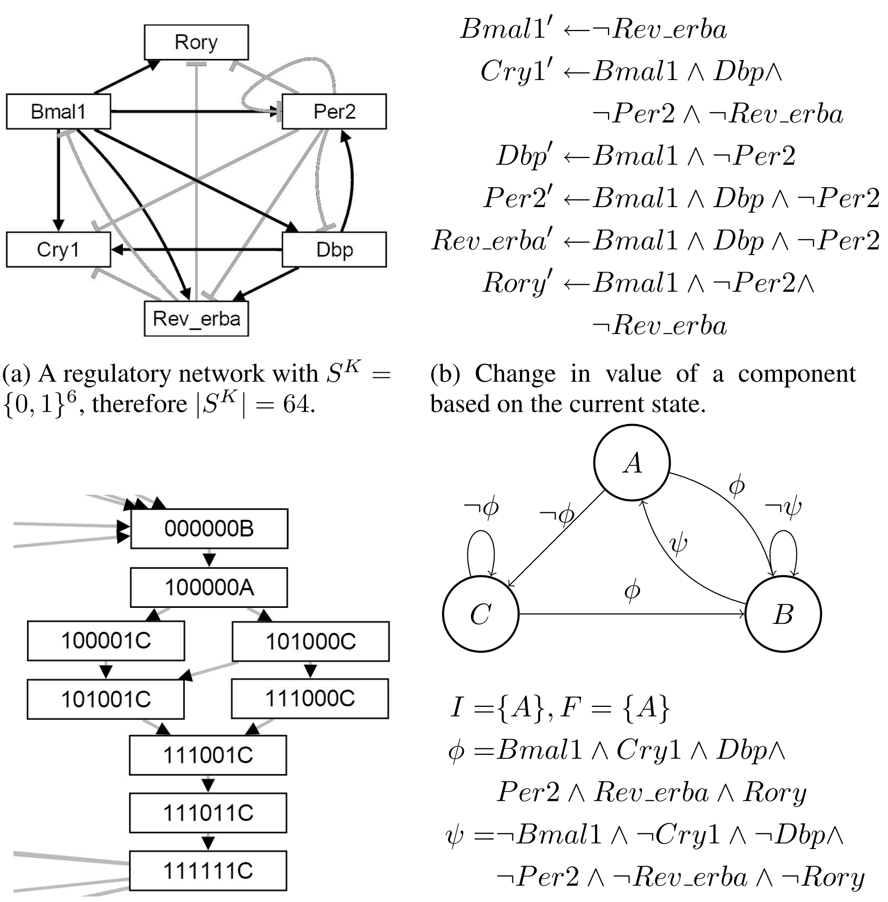
In this section we introduce the modelling framework [2] and present the relevant notions. The key concepts are illustrated in a case study in Figure 1, which models the mammalian circadian clock, based on results of [4].

In the logical modelling of biological components one abstracts from actual molecular concentrations and instead uses *activity levels*, each of which corresponds to a qualitatively different behaviour of a component. The system is described as a set V of named *components*, each of which has a *maximal activity level* assigned via a function $\rho : V \to \mathbb{N}$. The space of all configurations of a system is called *state space* and is denoted and defined as $S^K = \prod_{v \in V}[0, \rho(v)]$. The superscript K is used to later distinguish between different state sets. Note that an $s \in S^K$ is an integer-valued vector, therefore we use the notation s_v to refer to the value of $v \in V$ in the state s.

We focus on models with so-called *asynchronous update*, meaning that in each discrete state $s \in S$ we usually have multiple possible successors. The dynamics of the system—all the possible traces of a discrete simulation—can be in full described as an oriented graph, a so-called *Kripke structure* (KS) $K = (S^K, \to^K)$. Having the set S^K there are multiple ways of obtaining the relation $\to^K$, e.g. by setting logical equations, as illustrated in Figure 1b. Important for our purposes are the following characteristics of KS. For each $s \in S^K$ it is required that there is either only a loop $s \to^K s$ or each successor of s differs from it exactly by 1 in exactly one of the components. Denote $succ^K(s) = \{s' | s \to^K s'\}$ the set of successors of s in the relation $\to^K$, then for all $s \in S$ we have $succ^K(s) = \{s^1, \dots, s^j\}$ such that either $succ^K(s) = \{s\}$ or for each $i \in [1, j]$ there is an integer vector $e \in \mathbb{Z}^{|V|}$ s.t. $|e| = 1$ and $s^i = s + e$. Note that the length of one of e means that it has zeroes in all values except for exactly one which is in $\{-1, 1\}$.

2.1 Model Checking

To explore the behaviour of a system, we use the *model checking* [3] procedure, which allows to query whether a system has a certain property and obtain a *true/false* answer. Since we are usually interested in testing multiple KSs, each for a different regulatory function, we conduct multiple tests to obtain the set of the models for which the query resolves to *true*. We employ the Linear Temporal Logic (LTL) as the querying language and we resolve a query using the Büchi Automata [3] (BA) approach. In this method a property is described



$$\begin{aligned}
Bmal1' \leftarrow & \neg Rev\_erba \\
Cry1' \leftarrow & Bmal1 \wedge Dbp \wedge \\
& \neg Per2 \wedge \neg Rev\_erba \\
Dbp' \leftarrow & Bmal1 \wedge \neg Per2 \\
Per2' \leftarrow & Bmal1 \wedge Dbp \wedge \neg Per2 \\
Rev\_erba' \leftarrow & Bmal1 \wedge Dbp \wedge \neg Per2 \\
Rory' \leftarrow & Bmal1 \wedge \neg Per2 \wedge \\
& \neg Rev\_erba
\end{aligned}$$

(a) A regulatory network with $S^K = \{0,1\}^6$, therefore $|S^K| = 64$.

(b) Change in value of a component based on the current state.

$$\begin{aligned}
I = & \{A\}, F = \{A\} \\
\phi = & Bmal1 \wedge Cry1 \wedge Dbp \wedge \\
& Per2 \wedge Rev\_erba \wedge Rory \\
\psi = & \neg Bmal1 \wedge \neg Cry1 \wedge \neg Dbp \wedge \\
& \neg Per2 \wedge \neg Rev\_erba \wedge \neg Rory
\end{aligned}$$

(c) A excerpt from SW. Each node is a state with 6 values for the components, in the lexicographical order and one letter for a state of BA.

(d) A DBA encoding the requirement that the system must infinitely oscillate between states $(1,1,1,1,1,1) \in S^K$ and $(0,0,0,0,0,0) \in S^K$.

Figure 1: An illustrative study of the mammalian Circadian clock, based on [4]. The system (a) has the six boolean components $V = \{Bmal1, Cry1, Dbp, Per2, Rev\_erba, Rory\}$, all of which are required to oscillate (c) and for each $v \in V$ we have $\rho(v) = 1$. The behaviour is driven by the logical rules (b) and is partially depicted in (d).

as an oriented graph. Having this graph, a product with a KS describing a system is created. The query is *true iff* there exists a specific path in the product, as detailed below.

We describe the BA as a four-tuple $A = (S^A, \xrightarrow{\mathcal{L}(V)}, I^A, F^A)$, where $\mathcal{L}(V)$ is a language of propositional formulas [5] with the alphabet $\{v * n | v \in V, * \in \{\leq, \geq, <, >, =\}, n \in [0, \rho(v)]\}$ and $I^A \subseteq S^A, F^A \subseteq S^A$ are the sets of initial and final states respectively. To resolve whether a property described by automaton A holds in a structure K we create a *synchronous product* and search inside it for a so-called *lasso*, as further explained. The product $P = A \times K = (S, \rightarrow, I, F)$ is obtained as follows:

- $S = S^A \times S^K; I = I^A \times S^K; F = F^A \times S^K;$
- $(s^A, s^K) \rightarrow (r^A, r^K) \iff (s^K \rightarrow r^K) \wedge (s^A \xrightarrow{\phi} r^A) \wedge (s^K \models \phi),$

where $(s^K \models \phi)$ denotes the standard semantic consequence, i.e. the formula $\phi \in \mathcal{L}(V)$ is true in the state s^K. Having a product P the property encoded by automaton A is satisfiable in the structure K, denoted $\neg(K \models \neg A)$, *iff* there is a path $(s^1, \ldots, s^j, \ldots, s^k) \in (S)^j$ for any $j > 1$ such that $s^1 \in I$, $s^j \in F$, and $s^j = s^k$ [3]. This path is then called a *witness* of satisfaction of A by K in P. We denote W^P the set of all *witnesses* in P. Lastly we need to mention that in our approach we expect that many possible update functions may be considered and the check is run many times with only small changes in KS [1]. Subsequently we construct S explicitly and each of our algorithms is provided the total function $succ^P$ as an input. This is the core of the approach of [1], since we can test different regulatory functions by only making local, usually very small, changes to $succ^P$.

Throughout the article we distinguish between three sorts of BA—terminal BA, deterministic BA, and non-deterministic BA. The terminal BA (TBA) allow for encoding of *regular* properties [3], e.g. the time-series property [1]. The great simplification of these is that every final state has a loop, formally A is TBA *iff* for each $f \in F^A$ it holds that $f \xrightarrow{true} f$. We therefore know that a path exists whenever $s^j = s^k$ exists and only have to focus on searching for a path from s^1 to s^j. We also require a TBA to be deterministic and total. For a KS K over the set V and an automaton A of the formula $\phi \in \mathcal{L}(V)$ denote:

$$val(s^A) = (\{s^K \in S^K | s^A \xrightarrow{\phi} r^A, s^K \models \phi\})_{r^A \in S^A}.$$

A BA A is deterministic and total *iff* for each state of the system, exactly one of the labels is true, i.e. $\bigcap val(s^A) = \emptyset$ (deterministic) and $\bigcup val(s^A) = S^K$ (total). The advantage is that the number of transitions for each state in K

remains the same for each corresponding state in the product, since we always pair them with exactly one transition from A. A BA which is deterministic, but not terminal is called deterministic BA (DBA) and an extension of the approach from BA to DBA is the main contribution of this article. Later we also consider the non-deterministic BA (NBA), which allow to encode any LTL formula [3]. Our approach requires the transition function of an NBA to be total. However it can be easily shown that having a non-total NBA, we can just add a sink state with a loop and redirect all missing transitions there, allowing us to use any NBA.

2.2 Result Analysis

We start with the *cost* metric, which is based on the assumption that the all functions of a cell is optimized and changes to its state happen only if necessary. Consequently we are looking for *witnesses* with minimal number of steps, since these are most likely to represent an energy efficient behaviour. The *cost* is then the length, in states, of a shortest *witness*, defined as:

$$cost^P = min(\{k|(s^1, \dots, s^k) \in W^P\} \cup \infty).$$

Of special focus in our approach is the set of *witnesses* with minimal *cost*:

$$SW^P = \{(s^1, \dots, s^k)|cost^P = k \neq \infty\} \subseteq W^P,$$

which we also compute. While it can be argued that a single *witness* path is sufficient, having the set SW^P has additional advantages. First, it highlights areas with a strong requirement for order of changes in the system, as illustrated in Figure 1c. Second, we use the set SW^P to compute the *robustness* w.r.t. the property satisfaction. The *robustness* is an illustrative measure that states how likely it is that if we take a random walk of the length equal to the *cost*, we actually find a *witness*. This value creates an ordering on the set of models that satisfy a property, in which the results that have higher robustness, i.e. are more probable to meet the specification, may be considered as preferable. Note that there is no probability assigned to any transition, we therefore assume a uniform distribution, i.e. in a random walk the probability of taking any transition from a state s is given as one over the out-degree of s. For a path $(s^1, \dots, s^k) \in (S)^j$ we get the probability as:

$$prob(w) := \prod_{i=1}^{k-1} \frac{1}{deg^+(s^i)},$$

and since the probability of starting in any state is also uniformly distributed, the *robustness* is obtained as:

$$robustness^P := \frac{\sum_{w \in \mathrm{SW}^P}(prob(w))}{|I|}.$$

3 Case study

To illustrate our approach we present a short case study, based on the mammalian circadian clock model published in [4], depicted in Figure 1a. The logical rules in Figure 1b were obtained from the original differential equations in [4], where activation is converted as positive term, inhibition as negative, and multiplication as logical conjunction. Since in framework we do not model time explicitly, we do not consider the delays as in [4].

The system is expected to oscillate in all its components. We modelled the clock cycle as a requirement that the system oscillates between an initial state and a state that has the values of the components negated, see example in Figure 1d. Note that this could be also expressed as a time series, however we wanted to illustrate the use of DBA. Since we did not know the exact order of changes we tested all 64 options for the initial state. The *cost* is either 13, 15, or 17. This suggest that some of the configurations are harder to achieve. The *robustness* ranges between 0.000132 and 0 [†]. The highest robustness and the lowest cost was shared by the property in Figure 1d and the one where ϕ and ψ are swapped. This suggests that the most effective behaviour is to switch off all the components and then to switch them on again. Lastly, in Figure 1c we show an excerpt of the SW, showing the switch-on pathway. Note that the order of activations is highly deterministic and elucidates component dependencies. In particular we see that the order of components does not fully match the results of [4], e.g. *Dbp* must be activated before *Rev_erba*, as dictated by the logical function. The authors would suggest that this behaviour is due to absence of delays in our model and that in future this network should be remodelled using a boolean network with delays.

4 Algorithms

In this section we present the three algorithms that are used, in sequence, for the analyses of network dynamics. At first we show how to employ each of these algorithms with a property encoded by TBA. Later the extension to DBA is made. Lastly, we argue that the same extension can also be used with NBA and discuss the implications of non-determinism in terms of complexity and semantics. Due to lack of space we do not prove complexity and correctness, only show the key ideas of the proof. The main feature of our algorithms is that we exploit the *cost* value for practical performance. First please note several aspects of the problem:

[†] Robustness of 0 can occur due to representation of floating-point (rational) numbers in a computer, which we discuss in Section 4.3.

Algorithm 1 Calculate $Check(succ, I, F, full)$.

```
$X \leftarrow I, X' \leftarrow \emptyset, cost \leftarrow \infty, depth \leftarrow 1$
**for** $s \in S$ **do**
    $visit[s] \leftarrow \infty$
**end for**
**while** $X \neq 0$ **do**
    **for** $x \in X$ **do**
        $visit[x] = depth$
        **if** $(x \in F) \wedge (cost = \infty)$ **then**
            $cost \leftarrow depth$
        **end if**
        $X' \leftarrow X' \cup succ(x)$
    **end for**
    **if** $(cost \neq \infty) \wedge (\neg full)$ **then**
        $X \leftarrow \emptyset$
    **else**
        $X \leftarrow X' \cap \{x|visit[x] = \infty\}$
    **end if**
    $X' \leftarrow \emptyset, depth \leftarrow depth + 1$
**end while**
**return** $(cost, visit)$
```

- For a network with $|V|$ components, the out-degree of any state in a KS over V is at most $2 \cdot |V|$ since we can only increment or decrement by one in each dimension.

- Each component has at least 2 values, so $|S| \geq 2^{|V|}$ and $|V| \leq \log_2(|S|)$.

- The DBA guarantees that for each state of K there is only one edge allowed in A, therefore for $P = (S, \rightarrow, I, F)$ we have $| \rightarrow | \leq (2 \cdot |S| \cdot \log_2(|S|))$. For brevity we will further use $Size^P = 2 \cdot |S| \cdot \log_2(|S|)$.

- The product is fully constructed, i.e. the time complexity of any algorithm is at least $\mathcal{O}(Size^P)$. Also in practice we need $\mathcal{O}(Size^P)$ of memory to search through the graph.

- The transition system, if seen as a random process, has a Markov property. If we simulate the model by taking a random walk in K (or P), the the choice of following state depends only on the current state.

4.1 Property Checking

First we present the Algorithm 1 for model checking. Recall that we assume that the property was encoded by TBA, i.e. for a product $P = (S, \rightarrow, I, F)$ we only need to decide whether there is a path from I to F using the product successor function, $succ^P$. The algorithm is in its core a simple breadth-first-search (BFS), however it is modified so we obtain the *cost* value and also a labelling $visit : S \rightarrow \mathbb{N} \cup \infty$, that stores for each state the depth at which we visited the state. Note that if the shortest path has length k then $cost = k$ and for any $s \in S$ we have either $visit[s] \in [1, k]$ or $visit[s] = \infty$. Also note that we use an additional input parameter, called $full$. Currently we set it to $false$, however it will become useful in Section 4.4 for checking with DBA.

Proposition 4.1 *Algorithm 1 is correct. For* $P = A \times K = (S, \rightarrow, I, F)$ *it holds that* $(Check(succ^P, I, F, false))_1 \neq \infty \iff \neg(K \models \neg A)$.

Proof outline The algorithm is very similar to other implementations of BFS, e.g. [6] and consequently each state is visited at most once. The correctness of the labelling is then trivial.

Proposition 4.2 $SPACE(Check(succ^P, I, F, false)) \in \mathcal{O}(Size^P)$, $TIME(Check(succ^P, I, F, false)) \in \mathcal{O}(Size^P)$

Proof outline For the space we only use the labelling $visit$ for each state and store of size at most $|S|$, which can be done in $\log_2(|S|)$ space.

The time is given by the fact that we search from each state only once, looping through all the outgoing edges, so again at most $Size^P$. However note that if the *cost* is low we can exit the procedure early, examining only a subset of states, which is usually the case in practice.

4.2 Witness

To obtain a *witness*, we use a recursive depth-first-search (DFS), again modified for our purposes. Recall from Section 2.2 that we are looking for all the shortest paths. However, due to the strong non-determinism, the number of shortest paths grows exponentially w.r.t. *cost*. To prevent the exponential explosion in space and time complexity, we only store individual transitions that are on some $\omega \in SW^P$. Denote the set of transitions in SW^P as $SWT^P = \{(s^i, s^{i+1}) | (s^1, \ldots, s^i, s^{i+1}, \ldots, s^{cost}) \in SW^P\}$. Since the choice of successor is independent of all the previous steps and since all the paths in SW^P have the length of *cost*, we can reconstruct SW^P from SWT^P as $SW^P = \{(s^1, \ldots, s^{cost}) \mid \forall i \in [1, cost) : (s^i, s^{i+1}) \in SWT^P\}$.

To keep the complexity low and avoid searching through the paths we have already visited, we use altogether three distinct state labels. The $visit$ label is already provided by Algorithm 1. The $found$ label notes if a state lies on a known *witness* path and where. The $used$ label marks states that we already visited in DFS. Additionally we use the value $branch$ which points to the state where we branched from the last path that was found to be a shortest *witness* path. When a state is known to be part of a shortest *witness*, either by being a final state or by lying on some already known SW path, we just store the transitions from the last $branch$ to the current $depth$, avoiding duplicities.

Proposition 4.3 *Algorithm 3 is correct. If* $(cost^P, visit^P) \leftarrow Check(succ^P, I, F, false)$*, then* $Witness(succ^P, I, F, visit^P, cost^P) = SWT^P$.

Proof outline There are two parts to be proven. First, we need to show that the algorithm traverses through all the acyclic paths of length up to *cost*. In the algorithm we stop traversing in three cases. If the condition on line 1 is met, then we found the state in BFS sooner than now in DFS and therefore there must exist a shorter path to that state. If the condition on line 5 is met, then we found the *witness*. Lastly, if the condition on line 11 is met, then we either are at maximal depth or we already traversed from the state.

Second, we need to show that each transition is stored exactly once. When a new path is found we see that all transitions are stored on lines 6-9. At this point we set $branch$ to the current $depth$ and only decrement by one with each backtracking step. Therefore when storing transitions, we know that those in between 1 and $branch$ have been stored already. Also, when we hit a $found$ state, we know that all transitions up from that state have been stored already.

Proposition 4.4 $TIME(Witness(succ^P, I, F, visit^P, cost^P)) \in \mathcal{O}(Size^P)$ *and* $SPACE(Witness(succ^P, I, F, visit^P, cost^P)) \in \mathcal{O}(Size^P)$.

Proof outline The space is again simple—we only use space for states labelling, this time twice. For time complexity we again know that we do traverse any edge twice, since we label a state ($used$) after conducting a search from it and never search from it again.

4.3 Robustness

Lastly we focus on the *robustness* metric. For computation of *robustness* we utilize the set SWT^P as we know that only the transitions from the set lie on the shortest *witness* paths. Additionally we also know that since we use the shortest paths, there is no state that would be repeated on any of those

Algorithm 2 Calculate $DFS(x, depth, branch)$. The labellings $visit$, $found$, $used$, the sets Wit, F, the sequence $Path$, and the function $succ$ are shared between the recursive calls.

```
if $visit[x] < depth$ then
    return $branch$
end if
$Path[depth] \leftarrow x$
if $(x \in F) \vee (found[x] \leq depth)$ then
    for $d \in [branch, depth)$ do
        $Wit \leftarrow Wit \cup (Path[d], Path[d+1])$
        $found[x] \leftarrow depth$
    end for
    $branch \leftarrow depth$
else if $(depth < cost) \wedge (\neg used[x])$ then
    for $x' \in succ(x)$ do
        $branch \leftarrow min(DFS(x', depth+1, branch), depth)$
    end for
end if
$used[x] \leftarrow true$
return $branch$
```

paths. We can therefore simply descend through the set of shortest paths in a BFS manner, as we are sure that each state appears in only one iteration of the algorithm. Consequently the probability of reaching a state $s \in S$ in any of the shortest paths is equal to the probability of reaching it in $visit[s]$ steps, which is the invariant of the algorithm.

Proposition 4.5 *Algorithm 4 is correct. If* $(cost^P, visit^P) \leftarrow Check(succ^P, I, F, false)$, *and* $Wit^P \leftarrow Witness(succ^P, I, F, visit^P, cost^P)$. *Then* $Robustness(succ^P, I, F, cost^P, Wit^P) \simeq robustness^P$.

Algorithm 3 Calculate $Witness(succ, I, F, visit, cost)$.

```
for $s \in S$ do
    $found[s] \leftarrow \infty, used[s] \leftarrow false$
end for
$Wit \leftarrow \emptyset, Path \leftarrow (\bot)_{cost}$
for $i \in I$ do
    $DFS(i, 1, 1)$
end for
return $Wit$
```

Proof outline The invariant of the proof is that after k iterations, all the states up to the depth k are labelled with the reaching probability by any shortest *witness* path. Thus, after *cost* steps we have a labelling for all the final states at distance *cost* from I. Note that we state that the values are possibly only similar since we consider a possibility of having a slight rounding error for fractions for the sake of storage space.

Proposition 4.6 *$SPACE(Robustness(succ^P, I, F, cost^P, Wit^P)) \in \mathcal{O}(Size^P)$, $TIME(Robustness(succ^P, I, F, cost^P, Wit^P)) \in \mathcal{O}(Size^P)$.*

Proof outline Again we use just one label for all the states—in this case a fraction, which we store in at most $\log_2(|S|)$ space, having a possible rounding error in practice. Since we have no loops in SW^P, we certainly propagate from each state at most once through each edge, providing the bound of $|SWT^P| \leq Space^P$.

4.4 Extending to Deterministic BA

Up till now we have discussed usage of the algorithm for properties encoded by TBA. We can however extend the algorithms also to DBA, by stacking multiple calls of each of the algorithms. As explained in Section 2.1, to check for a property encoded by DBA, we are looking not only for a path from some $i \in I$ to some $f \in F$, but we also need a cycle containing f. We therefore need to first obtain the set $reach(F) \subseteq F$ of all reachable final states and then we need to decide whether there is a cycle on any $f \in reach(F)$. Also, previously we indicated that some of the advantages of the algorithm stem from the *witnesses* being acyclic, which does not hold any more as we are looking for a cycle on f. However we can break this cycle by creating a copy of f—a new state that has the same successors as f, but does not share its labels:

$$\forall s \in S : succ(s^{copy}) = succ(s) \land s^{copy} \notin S.$$

Lastly, in Algorithm 5 we denote $Wit[f^{copy}/f]$ the set of transitions where f^{copy} was replaced by f.

Proposition 4.7 *Algorithm 5 is correct. $Analyze(P) = (cost^P, SWT^P, robustness^P)$.*

Proof outline First note that we determine the *cost* already at lines 3-6. Later we are therefore only searching for paths that we already know are minimal. This is then done by joining shortest paths from I to f and from f to itself. Also for such f we know that its initial probability is given as probability of reaching it from I. Since in Algorithm 4 we set on line 4 the probability $prob[f] = \frac{1}{|\{f\}|} = 1$, we gain the final probability by multiplying the two.

Algorithm 4 Calculate $Robustness(succ, I, F, cost, Wit)$.

```
$X \leftarrow I, X' \leftarrow \emptyset, rob \leftarrow 0$
for $s \in S$ do
    if $s \in I$ then
        $prob[s] \leftarrow \frac{1}{|I|}$
    else
        $prob[s] \leftarrow 0$
    end if
end for
for $d \in [0, cost]$ do
    for $x \in X$ do
        $sw[x] \leftarrow \{x' | (x, x') \in Wit\}$
        $X' \leftarrow X' \cup sw[x]$
        for $x' \in sw[x]$ do
            $prob[x'] \leftarrow prob[x'] + \frac{prob[x]}{|Succ(x)|}$
        end for
    end for
    $X \leftarrow X'$
    $X' \leftarrow \emptyset$
end for
for $f \in F$ do
    $rob \leftarrow rob + prob[f]$
end for
return $rob$
```

Concerning the time complexity of the algorithm we can see that there is a stark increase w.r.t. the size of the set $reach(F)$. In the worst case the time is a square of what we had for TBA. Therefore if we expect a big $reach(F)$ set, one may probably want to trade the results provided by our analyses for performance gain of traditional model checking algorithms.

Proposition 4.8 $SPACE(Analyze(P)) \in \mathcal{O}(Size^P)$ *and* $TIME(Analyze(P)) \in \mathcal{O}(Size^P \cdot |F|))$.

Proof outline For the space we see that we keep results of at most two executions of $Check, Witness$, and $Robustness$ which is only a constant increase. Concerning the time we have two $Check$ executions for each of the $reach(F)$ members, with two executions of $Reach$ and $Witness$. These have again a time bound of $\mathcal{O}(Size^P)$, together $\mathcal{O}(Size^P \cdot |F|)$.

Algorithm 5 Calculate $Analyze(P)$ such that $P = A \times K = (S, \rightarrow, I, F)$ where A is a DBA.

```
$(cost\_reach, visit\_reach) \leftarrow Check(succ^P, I, F, true)$
$cost \leftarrow \infty$
for $f \in (F \cap \{s \in S | visit\_reach[s] \neq \infty\})$ do
    $(cost\_loop, visit\_loop) \leftarrow Check(succ^P, f^{copy}, f, false)$
    $cost \leftarrow min(cost, visit\_reach[f] + cost\_loop)$
end for
$Wit \leftarrow \emptyset, Rob \leftarrow 0$
for $f \in (F \cap \{s \in S | visit\_reach[s] \neq \infty\})$ do
    $(cost\_loop, visit\_loop) \leftarrow Check(succ^P, f^{copy}, f, false)$
    if $visit\_reach[f] + cost\_loop = cost$ then
        $Wit\_reach \leftarrow Wintess(succ, I, f, visit\_reach, cost\_reach)$
        $Wit\_loop \leftarrow Wintess(succ, f^{copy}, f, visit\_loop, cost\_loop)$
        $Rob\_reach \leftarrow Robustness(succ, I, f, cost\_reach, Wit\_reach)$
        $Rob\_loop \leftarrow Robustness(succ, f^{copy}, f, cost\_loop, Wit\_loop)$
        $Wit \leftarrow Wit \cup Wit\_reach \cup Wit\_loop[f^{copy}/f]$
        $Rob \leftarrow Rob + Rob\_reach \cdot Rob\_loop$
    end if
end for
return $(cost, Wit, Rob)$
```

4.5 Extending to Non-deterministic BA

Lastly we give a short comment on usage of NBA. There are two main practical differences between DBA and NBA. First, the complexity bound of each algorithm in terms of both space and time is dependent on the term $Size^P$. If A is however non-deterministic, it no longer holds for $P = A \times K = (S, \rightarrow, I, F)$ that $Size^P \leq 2 \cdot |S| \cdot \log_2(|S|)$. However we can easily adjust the complexity bounds by considering the maximal out-degree of any state in the DBA. More precisely, have an NBA $A = (S^A, \xrightarrow{\mathcal{L}(V)}, I^A, F^A)$ and denote $out(s^A) = \{(s^A, r^A) | (s^A, r^A) \in \xrightarrow{\mathcal{L}(V)}\}$. Then $Size^P \leq 2 \cdot |S| \cdot \log_2(|S|) \cdot max(\{out(s^A) | s^A \in S^A\})$, which we can readily use in all the previous complexity statements. The second, more elusive, difference is in the nature of *witness* and *robustness* analysis. While we still are correct w.r.t. to the definitions of Section 2.2, we do not have any longer one-to-one correspondence between transitions in S and S^K. Consequently, different encodings of one property can yield different *robustness*. For this problem we do not have any formal solution, and it should be taken into consideration by the user.

5 Conclusion

We revisited our previous results [1] that discussed usage of BFS model checking, *witness* search and a *robustness* metric, and provided novel, efficient algorithms for computing these. Moreover, we showed that two properties of our problem—having the transition system on the input and searching for the shortest *witnesses* of a property—provide advantages for the computation and can lead to a decrease in the complexity of the algorithms. Also, when using TBA-encodable properties, the complexity in both time and space is identical for all the algorithms, allowing for efficient pipelining.

Second, we showed how the algorithms can be used to extend the model checking procedure from simple reachability properties to the full LTL language, albeit under the cost of increase in time complexity. However as this cost is dependent on the set of reachable final states in an automaton, we argue that for properties that have a small reachable set of final states, e.g. a measurement is available, the increase in complexity is quite small in practice.

All the algorithms are experimentally implemented in the currently unpublished tool TREMPPI, a successor to the Parsybone tool [1]. The development version, which was used for the case study, is currently available at https://github.com/xstreck1/TREMPPI and is to be fully released in 2015.

References

[1] H. Klarner, A. Streck, D. Šafránek, J. Kolčák, and H. Siebert, "Parameter identification and model ranking of thomas networks," in *CMSB*, pp. 207–226, 2012.

[2] R. Thomas, "Regulatory networks seen as asynchronous automata: A logical description," *Journal of Theoretical Biology*, vol. 153, no. 1, pp. 1 – 23, 1991.

[3] C. Baier and J.-P. Katoen, *Principles of Model Checking*. The MIT Press, 2008.

[4] A. Korenčič, G. Bordyugov, D. Rozman, M. Goličnik, H. Herzel, *et al.*, "The interplay of cis-regulatory elements rules circadian rhythms in mouse liver," *PloS one*, vol. 7, no. 11, p. e46835, 2012.

[5] M. Huth and M. Ryan, *Logic in Computer Science: Modelling and reasoning about systems*. Cambridge University Press, 2004.

[6] S. S. Skiena, *The Algorithm Design Manual*. Springer Publishing Company, Incorporated, 2nd ed., 2008.

Design and simulation of a compact genetic flip-flop

Elise Rosati[1], Morgan Madec*[1], François Pêcheux[2], Yves Gendrault[1,3], Christophe Lallement[1], Jacques Haiech[4]

[1] Laboratoire des Sciences de l'Ingénieur, de l'Informatique et de l'Imagerie (ICube), UMR 7357, Equipe SMH, 300 Boulevard Sébastien Brandt, F-67412 Illkirch Cedex 02.

[2] Sorbonne Universités, UPMC Univ. Paris 06, UMR CNRS 7606, LIP6, F-75005, Paris.

[3] ECAM Strasbourg-Europe - 2 rue de Madrid, F-67300 Schiltigheim.

[4] Laboratoire d'Innovation Thérapeutique (LIT), UMR 7200, 74 route du Rhin, F-67400 Illkirch.

Abstract

Synthetic biology is an emerging research field at the interface between biotechnologies and engineering sciences with a unique potential. Inspired by electronics, our work deals with the design and the simulation of a biological flip-flop. Flip-flops are key devices for complex digital circuits for which signal synchronization is required. Unlike memories that can be obtained in biology with bistable constructs (*e.g.* a gene with a positive linear feedback), the state of a flip-flop can only change under the control of a particular signal called clock. We propose here a design made of three operons that achieve the same behavior as an electronic flip-flop. The construct is modeled and analyzed *in silico* through simulations. Results establish the proof of concept but also point out some constraints on model parameters which lead to specification about the strength of regulatory proteins and promoters that would be used in the actual system.

1 Introduction

Over the past fifteen years, synthetic biology, a new scientific field at the interface between biotechnologies and engineering sciences has developed rapidly. The goal of synthetic biology is to create new biological functions by assembling artificial or natural biological parts [1]. In this work, focus is put on a particular branch of synthetic biology: the design of new artificial genetic networks. The common philosophy adopted by the scientific community in this domain consists in designing these networks based on the principle of a construction game where complex biological function can be performed

*corresponding author: morgan.madec@unistra.fr

by assembling elementary building blocks, sometimes called BioBricks [2]. This way of thinking is similar to the one known in systems engineering; its potential is fully exploited in several domains such as digital electronics. On the other hand, it has been showed that at high level of abstraction the behavior of BioBricks can be described by Boolean equations [3]. From there was born the idea to build in the near future bio-computers with a computation core made of biological material, able to perform complex biological functions [4].

In microelectronics, digital circuits are favored in comparison to the analog ones because of several advantages [5]: digital circuits are more robust, less sensitive to noise, perturbations and performance alteration due to manufacturing process. Moreover, the design of such systems is simpler insofar as it is based on Boolean algebra and powerful computer-aided design (CAD) tools [6]. In synthetic biology, many BioBricks also exhibit a Boolean behavior and could benefit from it in the same way as digital electronic circuits. Thus, design of artificial genetic networks based on those BioBricks would be facilitated by the development of equivalent CAD tools adapted for synthetic biology, either from scratch [7], or on the basis of existing microelectronics tools [8]. The price to pay for the robustness and the relative design simplicity is an increase in the number of resources (building blocks and nets in electronics, genes and proteins in synthetic biology). For instance, an analog adder can be realized with a few tens of transistors while its digital equivalent requires over a hundred. In microelectronics, it is possible to integrate billions of transistors within a single chip. Genetic networks, even in the medium term, will probably never reach this degree of integration. Therefore, optimization of the number of resource is a critical issue.

Up to now, most of logic gates and basic combinatorial Boolean gates have already been realized with genetic networks [4]. However, these gates do not cover all the digital functions. For instance, sequential systems require memories, whose design is a trickier challenge. The first artificial constructs exhibiting a non-combinatorial behavior are the oscillators which switch between two steady states within a defined time interval [9]. In 2002, the toggle switches introduced by Gardner et al. [10] laid the foundation for the first genetic memory designed by Becskei et al. one year later [11]. This biodevice is based on a positive linear feedback. Alternatives to Becskei's biomemory have also been published in the early 2010's [12]. But this kind of memory is asynchronous whereas the computational design of complex sequential systems such as a finite state machine requires synchronous memories, called D-flip-flop (DFF) in electronics. In 2012, Hoteit et al. [13] developed the BioD, a biological DFF involving 7 operons and 10 proteins. BioD has been designed upon the standard structure used in microelectronics, *i.e.* two D-latches cascaded in a master-slave structure [5]. For this DFF, light acts as the clock signal by modulation of the effectiveness of a transcription factor. Note that equivalent behavior has also been reached using non-genetic systems (enzymatic reactions) [14]. In what follows, a new genetic network involving only 3 regulated operons is proposed. After a description of the concept, simulation results are shown. They are used to validate the concept and to challenge the robustness and the performance of the construct.

2 Description of the biological flip-flop

DFF is sensitive to two signals, namely Data (D) and Clock (Clk). By opposition with standard memories, the output of the DFF (Q), which should be bistable, may only change after rising (or falling) edges of the clock signal and according to the value of D: if D is high during the falling edge, Q turns (or stays) high whereas if D is low Q turns (or stays) low. We can transpose this behavior in the biological field, considering that "low" corresponds to a low concentration of a protein, that "high" corresponds to a concentration near the saturation concentration and that a falling edge is a sharp decrease of the concentration of a protein (by enhanced degradation or by inhibition). The structure of the biological DFF (BioDFF) we developed for the bacteria *Escherichia coli* is composed of 3 operons and involves 8 proteins and molecules. In addition, the system possesses two input signals and one reporter (GFP in this case). In this example, the D input can be 3-oxo-C12- homoserine lactone (3OC12HSL, thereafter termed AHL), which activates the LasR protein, which can in turn bind the *luxI* promoter and activate expression of operon #1. The oscillating Clk signal can be realized with pre-queuosine1 (PreQ1) and the adequate riboswitch [15]. The corresponding cartoon is given in Figure 1.

The operon #1 uses the *luxI* promoter ($\mathrm{P_{luxI}}$). Three proteins are synthesized when this operon is expressed: a repressor (CII), an enzyme (LasI) and an activator (CI). The enzyme LasI produces AHL (3OC12HSL). Upon binding with AHL, the protein LasR (expressed constitutively on a separate operon) activates the transcription of the luxI promoter on operon #1 [16]. LasI therefore plays the role of a self-activator. Direct addition of AHL, here Data signal, achieves the same effect. CII is a cross-repressor for operon #2 [13]. The phage λ regulator protein CI activates the transcription of operon #3 by binding to the $\mathrm{O_R}$ domains of its $\mathrm{P_{RM}}$ promoter [17]. The transcript of operon #1 contains a so-called riboswitch, a short RNA sequence located on the mRNA and sensitive to the presence of a specific ligand (PreQ1 in our system). Upon addition of the ligand, formation of a premature terminator structure on the mRNA arrests the transcription process [18]: the ligand act as a repressor of the genes located downstream of the riboswitch. The promoter of operon #1 also contains a LacI binding site.

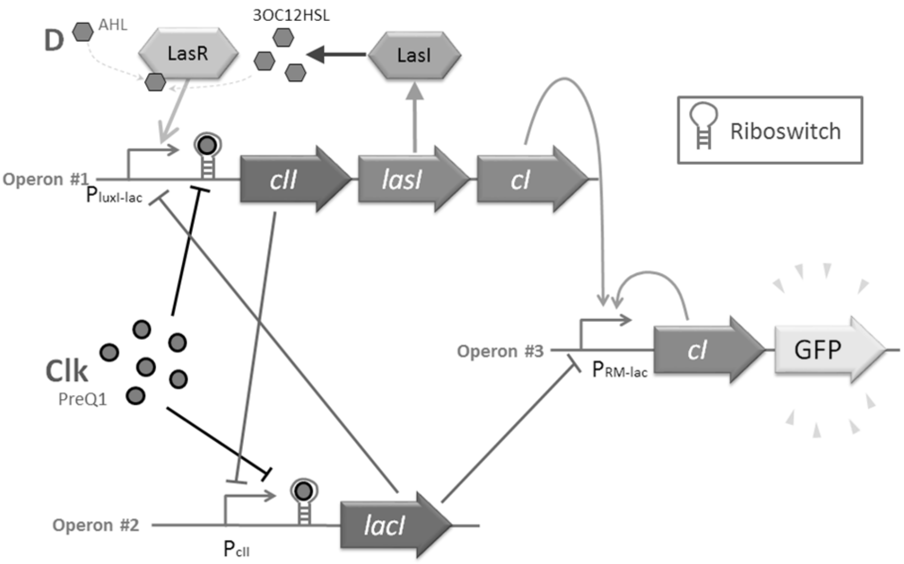


Figure 1: Cartoon of the biological D-flipflop involving 3 operons.

Hence, its expression is turned down in the presence of the repressor LacI. The expression of the operon #1 is controlled by the concentration of AHL, from the direct input or from the enzyme LasI, the molecule PreQ1 and LacI synthesized by the operon #2. The operon #2 synthesizes LacI, a repressor inhibiting the expression of operons #1 and #3. It has the same riboswitch sequence as operon #1 and is therefore also inhibited by PreQ1. Furthermore, operon #2 is sensitive to the repressor CII synthesized by operon #1. The operon #3 has a positive linear feedback construct [10], that is to say that it synthesizes a self-activator CI. Expression of the operon #3 is controlled by two regulating proteins: an activator synthesized by operon #1 and a repressor synthesized by operon #2.

The most obvious case is when PreQ1 is present. Operons #1 and #2 are repressed by PreQ1. As a consequence, the operon #3 is “insulated” from the rest of the circuit and acts as a bistable memory: if active, this activity is maintained by the synthesis of CI. If not active, there is no activator, thus it remains inactive. Now, let us consider the case where PreQ1 drops from high to low. At this point we have to distinguish between two cases. If AHL is high, the activity of both operons #1 and #2 increases: of operon #1 because of AHL and of operon #2 because of its constitutive promoter which is no more repressed. But at the same time, they start to synthesize cross-repressors

LacI and CII. Therefore, a race occurs between the two genes (*lacI* and *cII*). In order to reach the expected behavior, production of CII should be "boosted" in comparison with LacI. This could be achieved, for instance, by inserting two coding sequences for CII protein in the operon #1. This boost leads to the inhibition of the operon #2 and the activation of the operon #3 by CI. As a consequence, the output can rise to high. On the other hand, if AHL is low when PreQ1 drops from high to low, the scenario is more obvious because the operon #1, which does not have a constitutive promoter, remains inactive. steady state in which the operon #2 is active and represses operons #1 and #3 is quickly reached. As a consequence, the output activity drops to low. When PreQ1 is low, activity of the operon #2 depends only on activity of operon #1. If the operon #1 is inhibited, CII is not synthesized, thus the operon #2 is active and the output is low. If the operon #1 is active, the operon #2 is repressed and the output is high. Nevertheless, AHL may change at any time and especially may fall to low while PreQ1 is on. If the output is already inactive, it does not matter. However, to prevent a change of output state while it is active, the activity of the operon #1 has to be maintained by the self-activation of LasI. Finally, when PreQ1 rises from low to high, activity of operons #1 and #2 decreases without any modification of the output gene activity.

3 Modeling and simulation results

The BioDFF was modeled in SystemC-AMS [19] using a dedicated automated generator model [20], which produces a simulation-ready model from a gene netlist. Conventional equations for modelling the mechanisms of transcription of DNA into messenger RNA (mRNA) and the translation of mRNA into protein were used [21]:

$$\frac{d[mRNAn]}{dt} = \frac{k_{TRn}}{\left(1+\left(\frac{K_{An}}{\sum_p [Act_p]}\right)^{\alpha_n}\right)\cdot\left(1+\left(\frac{\sum_p [Rep_p]}{K_{Rn}}\right)^{\beta_n}\right)} - d_{mRNAn}\cdot[mRNAn]$$

$$\frac{d[Xn]}{dt} = k_{TLn}\cdot[mRNAn] - d_{Xn}\cdot[Xn]$$

where n is the operon number, k_{TR} is the transcription rate (default value is 0.1), k_{TL} is the translation rate (0.1 by default), K_A and K_A are the affinity of activators and repressors on the promoter (resp. 0.2 and 0.01 by default), α and β are the Hill's coefficient for the activator and for the repressor respectively (2 by default) and d_x and d_{mRNA} are the degradation rate of proteins and mRNA (0.1 by default). Concentration are normalized by the concentration at saturation of the proteins in the cell (*i.e.* concentration vary between 0 and 1).

For the three operons, all the 20 parameters are set to the default value except K_{R2}, which is equal to the half of the default value (0.005) and models the fact that CII is boosted in comparison to LacI. Parameter values are chosen accordingly to [22].

A transient simulation (*i.e.* time course) was performed with a testbench for Clk and D that covers all the possible scenarii. The simulation results are given in Figure 2. The expression of the three operons as well as the concentration of reporter corresponded to the expected signals. In particular, it was observed that output changes occur only on falling edges of the clock signal and in accordance with the presence or absence of D at the edge. Glitches were observed on falling edge of Clk when D and Q are high. They can be explained by a loss of gene expression during the race between operon #1 and #2 described previously.

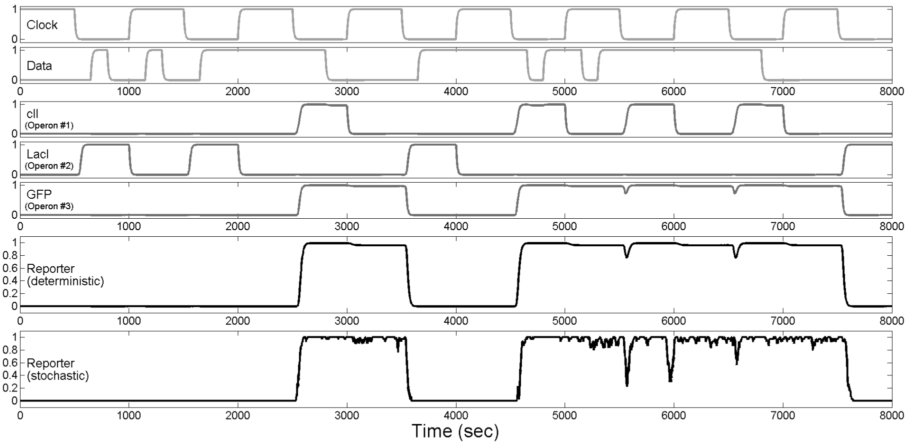
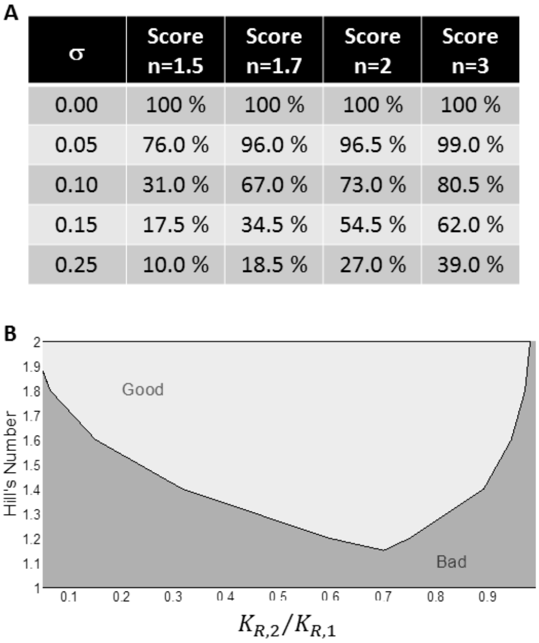


Figure 2: Simulation results of the biological D-flipflop with default parameters. Simulations results for the reporter are provided both by a deterministic simulator (third line) and a stochastic simulator (fourth line).

Deterministic simulation gives a proof of concept of our construct. Nevertheless, further analyses have to be carried out in order to check its robustness. Thereafter, two of them are described. The first one aims at estimating the noise immunity of the system. The second one deals with the sensitivity of the response with the variations of model's parameters.

First, noise immunity is discussed. The cornerstone of the system's operation is the race between CII and LacI at the falling clock edge. What would happen if these two proteins existed only in very small quantities? To answer this question, stochastic simulations were carried out with an algorithm derivated from Gillespie's one [23]. Results are also given in Figure 2. It can be pointed out that the expected response of the system was reached even if the number of molecules is low.



σ	Score n=1.5	Score n=1.7	Score n=2	Score n=3
0.00	100 %	100 %	100 %	100 %
0.05	76.0 %	96.0 %	96.5 %	99.0 %
0.10	31.0 %	67.0 %	73.0 %	80.5 %
0.15	17.5 %	34.5 %	54.5 %	62.0 %
0.25	10.0 %	18.5 %	27.0 %	39.0 %

Figure 3: Results of the robustness test. (A) shows the score (i.e. the percentage of output waveforms that looks like the expected one from a digital point of view only over the 1,000 parameters set) as a function of the parameters spread and Hill's number. (B) gives the range of the $K_{R,2}$ over $K_{R,1}$ ratio for which the systems works as a function of Hill's number.

The robustness of the system was analyzed through Monte-Carlo simulations. First, the parameters were altered one by one in order to identify the most critical of them. As expected, parameters related to the repression of operon #1 by LacI and operon #2 by CII were the most critical. Then, the influence of the repressor-promoter binding cooperation factor (n_1 and n_2) was addressed. For this simulation process, parameters $K_{TR,n}, K_{TL,n}, K_{A,n}$ and $K_{R,n}$ are allowed to vary around their default value. Let Λ be the vector containing these parameters. A test consists of 1,000 random draw ($\Lambda_1, ..., \Lambda_{1000}$) in which each set is computed as following: $\Lambda_k = \Lambda_0 \cdot 10^{1+\sigma \cdot \Psi}$, where Ψ is a vector with 10 random values drawn with a standard normal distribution and σ is also a vector containing the standard-deviation of the spread for each parameters. Results for different Hill's numbers and different values of σ are compiled in Figure 3A. The score corresponds to the percentage of output waveforms that looks like the expected one (from a digital point of view only) over the 1,000 parameters set. We can point out that Hill's number has a high influence on the robustness of the system.

This result is confirmed by Figure 3B giving for each Hill's number the possible range of variation of the $K_{TR,2}$ versus $K_{TR,1}$ ratio. Again, the higher the Hill's number, the wider the range and the bigger the robustness.

4 Conclusion

To conclude, a new genetic D flip-flop has been designed. It is an alternative of a structure previously introduced by Hoteit *et al.* in 2012. Its main advantage consists in being more compact (only three operons against seven for Hoteit). The other side of the coin is that our system relies on a competitive reaction between two repressors and a corresponding associated promoter, which renders the system less robust than Hoteit's. Indeed, the simulations carried out in SystemC-AMS showed that system parameters should be controled precisely enough to ensure proper functioning of the system, which can be easily done in simulations but could be a tricky challenge when realized with actual genetic material.

References

[1] D. Endy, Foundations for engineering biology., Nature, vol. 438, no. 7067, pp. 449-53, Nov. 2005.

[2] B. Canton, A. Labno, and D. Endy, Refinement and standardization of synthetic biological parts and devices., Nat. *Biotechnol.*, vol. 26, no. 7, pp. 787-93, Jul. 2008.

[3] R. Weiss, G. E. Homsy, and T. F. Knight Jr, Toward in vivo digital circuits, in Evolution *as Computation, DIMACS Workshop*, L. Landweber and E. Winfree, Eds. Springer, 2002, pp. 275-295.

[4] S. Ausländer, D. Ausländer, M. Müller, M. Wieland, and M. Fussenegger, Programmable single-cell mammalian biocomputers., *Nature*, vol. 487, no. 7405, pp. 123-7, Jul. 2012.

[5] Paul Horowitz and W. Hill, *The Art of Electronics.* Cambridge University Press, 1989.

[6] L. Lavagno, G. Martin, and L. Scheffer, Electronic Design Automation for Integrated Circuits Handbook - 2 Volume Set, Apr. 2006.

[7] J. Beal, R. Weiss, D. Densmore, A. Adler, E. Appleton, J. Babb, S. Bhatia, N. Davidsohn, T. Haddock, J. Loyall, R. Schantz, V. Vasilev, and F. Yaman, An end-to-end workflow for engineering of biological networks from high-level specifications., ACS *Synth. Biol.*, vol. 1, no. 8, pp. 317-31, Aug. 2012.

[8] Y . Gendrault, M. Madec, C. Lallement, and J. Haiech, Modeling biology with HDL languages: A first step toward a genetic design automation tool inspired from microelectronics, IEEE *Trans. Biomed. Eng.*, vol. 61, no. 4, pp. 1231-1240, 2014.

[9] M. B. Elowitz and S. Leibler, A synthetic oscillatory network of transcriptional regulators., Nature, vol. 403, no. 6767, pp. 335-8, Jan. 2000.

[10] T. S. Gardner, C. R. Cantor, and J. J. Collins, Construction of a genetic toggle switch in Escherichia coli., Nature, vol. 403, no. 6767, pp. 339-42, Jan. 2000.

[11] A. Becskei, B. Séraphin, and L. Serrano, Positive feedback in eukaryotic gene networks: cell differentiation by graded to binary response conversion., EMBO *J.*, vol. 20, no. 10, pp. 2528-35, May 2001.

[12] D.-E. Chang, S. Leung, M. R. Atkinson, A. Reifler, D. Forger, and A. J. Ninfa, Building biological memory by linking positive feedback loops., Proc. *Natl. Acad. Sci. U. S. A.*, vol. 107, no. 1, pp. 175-80, Jan. 2010.

[13] I. Hoteit, N. Kharma, and L. V arin, Computational simulation of a gene regulatory network implementing an extendable synchronous single-input delay flip-flop., Biosystems., vol. 109, no. 1, pp. 57-71, Jul. 2012.

[14] K. MacVittie, J. Halámek, and E. Katz, Enzyme-based D-flip-flop memory system., Chem. *Commun. (Camb).*, vol. 48, no. 96, pp. 11742-4, Dec. 2012.

[15] W. C. Winkler and R. R. Breaker, Regulation of bacterial gene expression by riboswitches., Annu. *Rev. Microbiol.*, vol. 59, pp. 487-517, Jan. 2005.

[16] K. M. Gray, L. Passador, B. H. Iglewski, and E. P. Greenberg, Interchangeability and specificity of components from the quorum- sensing regulatory systems of Vibrio fischeri and Pseudomonas aeruginosa., J. *Bacteriol.*, vol. 176, no. 10, pp. 3076-80, May 1994.

[17] D. L. Court, A. B. Oppenheim, and S. L. Adhya, A new look at bacteriophage lambda genetic networks., J. *Bacteriol.*, vol. 189, no. 2, pp. 298-304, Jan. 2007.

[18] Z. Gong, Y. Zhao, C. Chen, and Y. Xiao, Computational study of unfolding and regulation mechanism of preQ1 riboswitches., PLoS *One*, vol. 7, no. 9, p. e45239, Jan. 2012.

[19] A. Vachoux, C. Grimm, and K. Einwich, Analog and Mixed Signal Modelling with SystemC-AMS, in IEEE *International Symposium on Circuits and Systems (ISCAS)*, 2003.

[20] M. Madec, F. Pecheux, Y. Gendrault, L. Bauer, J. Haiech, and C. Lallement, EDA inspired open-source framework for synthetic biology, in 2013 *IEEE Biomedical Circuits and Systems Conference, BioCAS 2013*, 2013, pp. 374-377.

[21] Y. Gendrault, M. Madec, C. Lallement, F. Pecheux, and J. Haiech, Synthetic biology methodology and model refinement based on microelectronic modeling tools and languages., Biotechnol. *J.*, vol. 6, no. 7, pp. 796-806, Jul. 2011.

[22] U. Alon, *An Introduction to Systems Biology: Design Principles of Biological Circuits.* 2006, p. 320.

[23] D. T. Gillespie, Exact Stochastic Simulation of Coupled Chemical Reactions, J. *Phys. Chem.*, vol. 93555, no. 1, pp. 2340-2361, 1977.

Modelling *Dictyostelium discoideum* aggregation through a discrete excitability model with directional sensing

Sofia Almeida[1,2,3] and Rui Dilão[3]

[1] Laboratory of Excellence Labex SIGNALIFE "Network for Innovation on signal Transduction Pathways in Life Sciences", Grant ANR-11-LABX-0028-01.
[2] Inria Sophia Antipolis Méditerranée, 2004, route des Lucioles - BP 93 06902 Sophia Antipolis Cedex, France.
[3] University of Lisbon, Instituto Superior Técnico, GDNL, Av Rovisco Pais, 1049-001 Lisbon, Portugal.

Abstract

The social amoebae of *Dictyostelium discoideum* (*Dd*) are unicellular organisms that, under prolongated starvation, aggregate through a reaction-diffusion signalling system and later differentiate to form a pluricellular organism. In this work, the Kessler-Levine simple discrete model for *Dd* early-stage aggregation is extended to include the Dilão-Hauser directional sensing effect. The resulting model describes all the known patterns of *Dd* aggregation, which include the spontaneous formation of cAMP self-sustained target and spiral waves and streaming effects. One alteration on this model leads to the emergence of self-organising complex regular patterns. The bifurcation analysis of the main processes has been performed.

1 Introduction

The social amoebae *Dictyostelium discoideum* usually live as simple individual organisms in the soil leaf litter, feeding on bacteria and dividing by mitosis. Under starvation, however, they aggregate to form a pluricelular life form, [4]. A colony of aggregating *Dd* cells constitutes an interesting system for the study of pattern formation, which is the purpose of this work.

Starvation induces cells to produce the chemical compound cAMP (cyclic adenosine monophosphate) and also a phosphodiesterase enzyme PDE that degrades it. cAMP is relayed in the medium in an oscillatory manner and propagates as a reaction-diffusion wave, very often a spiral. The amoebae move chemotactically in the direction of the gradient of cAMP concentration, forming a very particular pattern called streaming — a ramified network that converges into the cAMP wave diffusive center, where they eventually aggregate. From this aggregate they start to climb vertically, passing by several stages of morphogenesis and undergo cellular differentiation between two types of cells (stalk cells and spore cells), culminating in a mature pluricelular organism — the fruiting body [5].

Dictyostelium aggregation has been extensively modeled throughout the years and there are a variety of models to describe cAMP production and relaying dynamics, [6], [7] and [8], spiral wave break-up, [9], [10] and [11], amoebae movement and stream formation, [12], [13] and [14], *etc*. The model developed in this work is constructed from the merging of two models, the Kessler-Levine model for cAMP production, [1], and the Dilão-Hauser model of directional sensing, [2], and is thoroughly explored for several values of the parameters.

2 Models and Methods

To account for the amoebae's sensing and the production of cAMP, the model proposed by Kessler and Levine was used, [1]. This is a simple, discrete, model that doesn't incorporate the majority of biochemical features and machinery of amoeba cells, but is, nevertheless, successful in reproducing streaming patterns. In this model, amoebae are interpreted as "bions": "simple elements that mimic the cell's behaviour by a set of simple, easily computable rules", [1]. Each amoeba has an internal state that represents the availability of cAMP receptor sites, accordingly:

i) State 0: amoebae are excitable. They are not emitting cAMP, but they detect its local concentration c and if it is above the treshold c_T they become excited, changing to state 1.

ii) State 1: amoebae are excited. They emit a fixed amount Δc of cAMP over τ time units. After τ they progress to state 2.

iii) State 2: amoebae are quiescent. They neither emit cAMP nor can be further excited during t_R time units. After t_R they revert to state 0.

These dynamic rules are successful in reproducing the auto-excitable behaviour of the system, and a propagating target wave results from the amoebae amplification of the signal emitted by a localized temporal "pacemaker", oscillating periodically around c_T, [1].

The model of Kessler-Levine just exposed is an early-stage aggregation model of *Dictyostelium* as it successfully reproduces the streaming pattern but has no aggregation in the diffusive center. In this work, we will use the same cAMP production dynamics, with the three defined amoebae internal states. As in the Kessler-Levine model, the propagation of cAMP in the medium is given by the reaction-diffusion equation:

$$\frac{\partial c}{\partial t} = D\Delta c - \Gamma c + (sources), \qquad (1)$$

where c is the cAMP concentration, D is the diffusion coefficient and Γ is the decay rate representing the degradation of cAMP due to phosphodiesterase activity or natural degradation. The source term represents the local contributions of cAMP by the amoebae.

Another important feature of this work is the directional sensing operator derived by Dilão-Hauser, [2]. They have proposed that the amoebae are sensitive to the direction of propagation of the cAMP wave. This is justified by the fact that amoebae sense and follow an oscillatory gradient, but their movement is not oscillatory — chemotactic wave paradox. These authors suggested that the amoebae only move when the cAMP gradient is in the opposite direction of the wave propagation, which they then derive to be equivalent to the verification of the condition:

$$\text{signal}\left(\frac{\partial c}{\partial t}\right) > 0. \tag{2}$$

They have tested this directional sensing condition using an external Ginzburg-Landau reaction-diffusion field model.

In this work, the two models described above were merged. The goal is to reproduce the phenomena of pattern formation and aggregation in *Dictyostelium dicoideum* colonies and to understand the basic mechanisms underlying the observed phenomena.

The integration of the reaction-diffusion equation was done using a method proposed by Dilão-Sainhas, [3]. They noted that space and time scales (Δx and Δt) are not independent in diffusion and reaction-diffusion systems and have found that the relation between space and time scales that minimizes integration errors is

$$\frac{D\Delta t}{(\Delta x)^2} = \frac{1}{6}. \tag{3}$$

Furthermore, they have proposed a new class of explicit difference methods for the integration of reaction-diffusion equations in 1, 2 and 3 dimensions, where the dependence of time and space scales occurs naturally. Denoting by $v_{i,j}^k$ the concentration of cAMP at given lattice cell with coordinates (i, j), at a given instant of discretized time k, the 2D Dilão-Sainhas difference method used in this work is

$$\begin{aligned} v_{i,j}^{k+1} &= v_{i,j}^k + \frac{1}{9}(v_{i-1,j}^k + v_{i+1,j}^k + v_{i,j-1}^k + v_{i,j+1}^k - 4v_{i,j}^k) \\ &+ \frac{1}{36}(v_{i-1,j-1}^k + v_{i+1,j+1}^k + v_{i+1,j-1}^k + v_{i-1,j+1}^k - 4v_{i,j}^k) \\ &+ \Delta t f(v_{i,j}^k), \end{aligned} \tag{4}$$

where, by (1), $f(x) = -\Gamma x$.

In our simulations, we randomly distribute the amoebae in a 2D circle inside a 200×200 square lattice of cell side length Δx. Each amoebae is represented by their center of mass and occupies no area. The cAMP production follows the dynamics proposed by Kessler-Levine and the movement of the amoebae obeys the directional sensing condition derived by Dilão-Hauser. Eq. (1) describes the propagation of the cAMP relayed in the medium and is integrated in the circle using the Dilão-Sainhas method (4), with no flux boundary conditions.

3 Results and Discussion

3.1 Propagation of cAMP with a pacemaker source; calibration

The parameters of the model have the following reference values: $\Delta c = 150\ nmol.mm^{-2}$, $c_T = 0.5\ nmol.mm^{-2}$, $t_R = 20\ s$, $\tau = 0.2\ s$ and $\Gamma = 0.5\ s^{-1}$. These were found experimentally to garantee the entrance in oscillatory regime, it is kept a relation of $\frac{\Delta c}{c_T} = 300$ as proposed in Kessler-Levine [1]. The diffusion coefficient is hidden into the scaling condition (3), but will be estimated from the experimental data (see below). The integration time step used in the integration method (4) is $\Delta t = 0.1\ s$.

In the middle of the circular domain, we place a pacemaker (imposed source of cAMP) oscillating around c_T, with amplitude $A = 0.4\, nmol.mm^{-2}$ and period $T = 30\ s$ ($c = c_T + A\cos(2\pi t/T)$). The pacemaker acts as a diffusion center and the production of cAMP by the amoebae together with its degradation according to (1) leads to the propagation of cAMP in the 2D media as a reaction-diffusion target wave.

It was found that a minimum amoebae density of 27 $\%$ guarantees wave propagation and that the parameter c_T influences the velocity of waves, with waves travelling faster for smaller c_T. For $c_T = 0.5\ nmol.mm^{-2}$, the wavefront takes around 30 s to propagate one radius of the circular domain, or half the size of the lattice length 100 Δx. The system can now be calibrated, from the known velocities of propagation of diffusion waves in aggregating *Dd* colonies of 300 $\mu m.min^{-1}$, [1]. So, from this value, we obtain $\Delta x = 1.5\ \mu m$, which represents the spatial scale of the system. Now using eq. (3), we may obtain the diffusion coefficient of $D = 3.75\ \mu m^2.s^{-1}$, that characterizes this system.

3.2 Propagating cAMP and amoeba motion with directional sensing

We now set the amoebae in motion, allowing them to move only when they are on state “1”, as it is done in Kessler-Levine [1]. A parameter v defines the value of the velocity of each individual amoeba in the $\overrightarrow{x}$ and $\overrightarrow{y}$ directions.

The amoeba moves in the direction of the gradient of cAMP if condition (2) is verified. In the simulations, we have observed the emergence of transient streaming currents culminating with the aggregation of all the amoebae in the diffusive center, i.e., where the pacemaker is located. In the supplementary video 1 available at http://sd.tecnico.ulisboa.pt/NonLinear_Dynamics_Group is possible to see what happens to 16000 amoebae moving with $v = 20\ nm.s^{-1}$.

The model reproduces successfully the streaming and aggregation of *Dictyostelium* colonies. The parameters that determine the dynamics of the amoebae are the velocity v, the degradation rate Γ and the relaxation time t_R (which corresponds approximately to the period of wave emission). A diagram for t_R and v, representing the transient states of the amoebae colonies, is shown in fig. 1. To provide a general overview of the system, the diagram covers a large domain of parameters in a logarithmic scale. Blue corresponds to low concentration of cAMP and red to high concentration. Amoebae are represented in green. These diagrams have been calculated with a colony of 8 000 amoeba.

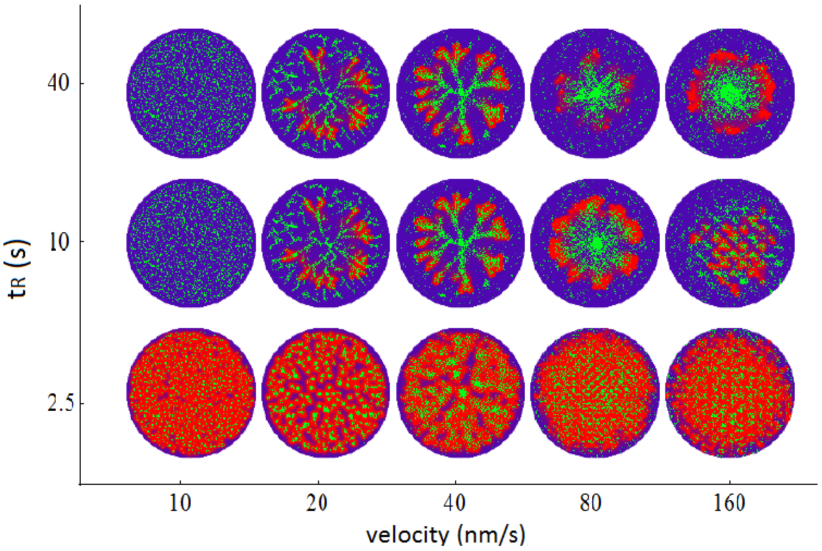


Figure 1: Diagram for the transient states of amoeba colonies, as a function of v and t_R. The amoebae velocity axis is a logarithmic scale of base 2 and the t_R axis a logarithmic scale of base 4. Parameters are $\Gamma = 0.5\ s^{-1}$, $\tau = 0.2\ s$, $\Delta c = 150\ nmol.mm^{-2}$ and $c_T = 0.5\ nmol.mm^{-2}$. The images in the figure were obtained for the parameter values $v = 10, 20, 40, 80, 160\ nm.s^{-1}$ and $t_R = 2.5, 10, 40\ s$.

For high oscillatory frequencies, low t_R, there is no streaming or aggregation near the cAMP source and the amoebae form small aggregates along the way. For higher t_R, there is a streaming zone, where an increase in the value of v provokes the decreasing of the number of streams, which in turn become thicker and less defined, until streaming is no longer observed and the system enters a zone where there is aggregation without streaming. A relatively high value of t_R seems to be required to obtain aggregation in the pacemaker. Moreover, increasing the number of amoebae N also interferes with the dynamics, resulting, generally, in an increase of the number of streams.

Now, fixing $t_R = 20\ s$ and keeping amoebae velocity v in the streaming zone, the effects of Γ together with v are shown in the diagram of fig. 2.

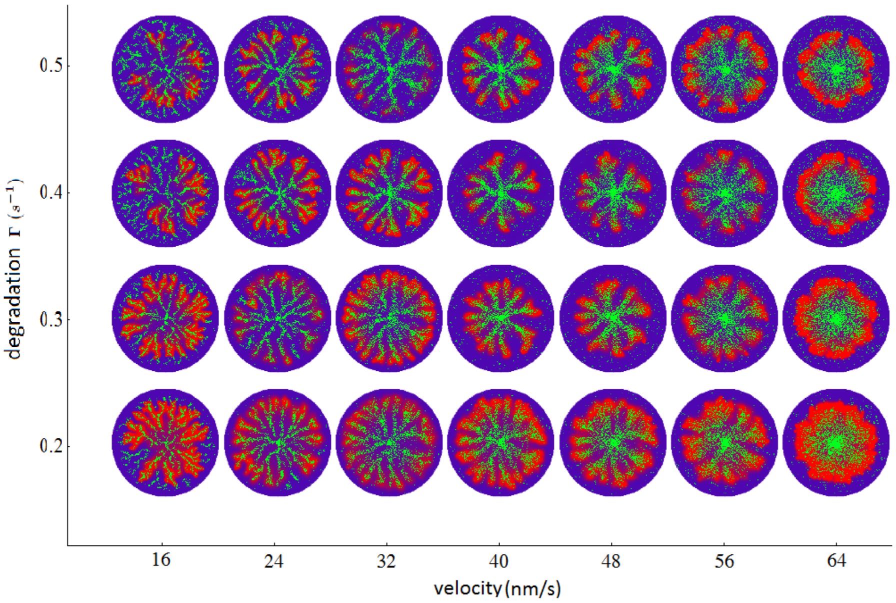


Figure 2: Diagram for the transient states of amoeba colonies, as a function of v and Γ — streaming zone. Parameters are $t_R = 20\ s$, $\tau = 0.2\ s$, $\Delta c = 150\ nmol.mm^{-2}$ and $c_T = 0.5\ nmol.mm^{-2}$. The images in the figure were obtained for the parameter values $v = 16, 24, 32, 40, 48, 56, 64\ nm.s^{-1}$ and $\Gamma = 0.2, 0.3, 0.4, 0.5\ s^{-1}$.

In the simulations in fig. 2, the effects of the variation of v on the system are the ones already discussed concerning the number and the thickness of the streams. On the other hand, lowering Γ also results in the formation of more and thinner streams, although this effect is not so pronounced. For $\Gamma \geq 0.6\ s^{-1}$, wave propagation doesn't occur anymore.

It is important to note that the patterns obtained in figures 1 and 2 resemble some results obtained from experimental studies, [15] and [16]. For instance, Kriebel et al., [15], observed a streamless aggregation in *Dd* deletion mutants for the production of the protein adenyl cyclase (ACA) and the formation of several small aggregates, similar to what may be observed in certain images of fig. 1, for *Dd* mutants with a different ACA distribution than the wild type. Moreover, Hilgardt et al., (2008), [16], also obtain several smaller aggregation territories of *Dd* with the introduction of IPA in the medium. They, in fact, report a wider range of cAMP wave frequencies observed in their experiment [16].

The parameters that here were found here that have influence on the system, are very often overlooked in experimental research. Amoebae velocity in particular is hardly ever addressed. However, here we have found that the frequency of cAMP emission and the amoebae velocity together with degradation rate are at the core of the change in dynamics in aggregating *Dd* colonies. This simple model reproduces streaming and aggregation, allowing a straightforward understanding of the essential aspects of pattern formation.

3.3 The emergence of spiral dynamics

Moreover, it is also possible to obtain spontaneously formed spirals without an imposed pacemaker. This is another extension to the model that better approximates to what happens in nature.

With no imposed cAMP pacemaker source, an initial condition of constant cAMP concentration c_0 was given to each amoeba. As the amoebae only start emitting if the local concentration of cAMP is above the treshold c_T, we set $c_0 < c_T$, so that only the spatial regions with high enough number of amoebae will have a cAMP concentration above threshold. These places act as diffusion centers for a first travelling wave and, afterwards, one or more spontaneously formed diffusive centers may emerge, forming self-sustained spirals, one of the most common type of wave obtained. The position of second centers differs from the position of the initial ones. This process might be seen in fig. 3, for 8000 fixed amoebae. Supplementary video 2 shows the same simulation with more detail (http://sd.tecnico.ulisboa.pt/NonLinear_Dynamics_Group/Videos).

The bifurcation parameters for the emergence and positioning of the diffusive centers are c_0 and amoebae distribution.

Furthermore, in fig. 4, simulations are made for five different initial amoebae distributions. Here it can be seen that the bifurcating parameters c_0 and amoebae positioning influence the emergence, the number, the type (target, spiral or interference formations) and the position of the self-sustained diffusion centers. These parameters reflect directly on the intial spatial distribution of cAMP, that here is seen as the essential feature behind the emergence (or

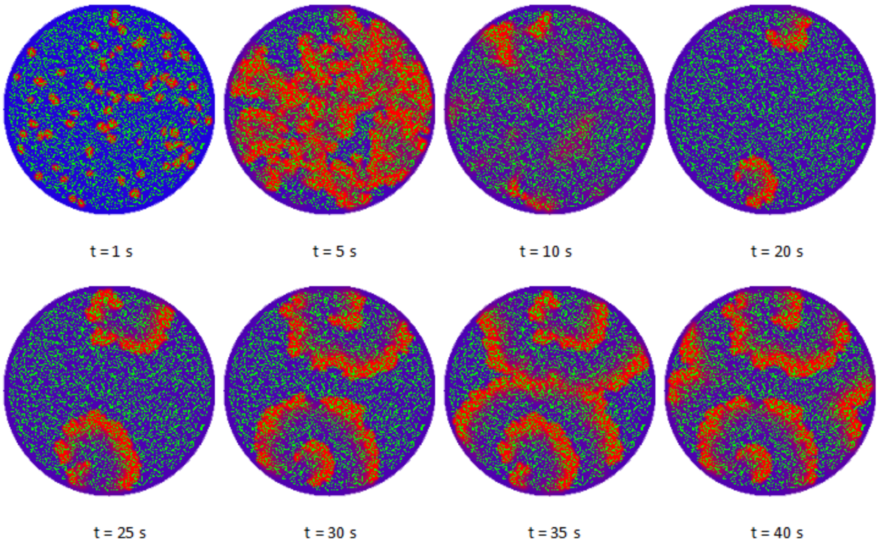


Figure 3: Spontaneous spiral wave generation in the course of time. An initial cAMP concentration $c_0 = 0.2 \; nmol.mm^{-2}$ was given per amoeba. As time progresses, two self-sustained rotating spiral waves emerge. The parameters of the simulation are $\tau = 0.4 \; s$, $t_r = 10 \; s$, $\Gamma = 0.5 \; s^{-1}$, $c_T = 0.5 \; nmol.mm^{-2}$ and $\Delta c = 300 \; nmol.mm^{-2}$.

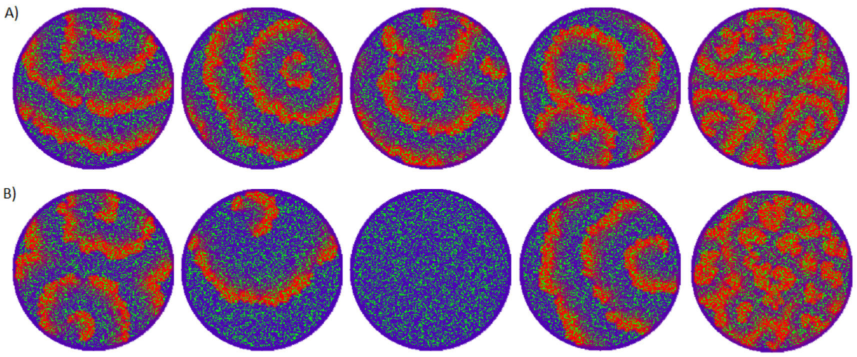


Figure 4: Spiral waves for five different initial amoebae distributions. The number, the type and the position of the self-sustained diffusion centers change: A) $c_0 = 0.15 \; nmol.mm^{-2}$; B) $c_0 = 0.2 \; nmol.mm^{-2}$. The other simulation parameters are $\tau = 0.4 \; s$, $t_r = 10 \; s$ ($t_r = 7 \; s$ for the fifth case), $\Gamma = 0.5 \; s^{-1}$, $c_T = 0.5 \; nmol.mm^{-2}$ and $\Delta c = 300 \; nmol.mm^{-2}$.

not) of self-sustained diffusive centers.

Now the amoebae are set in motion, after the establishment of the spirals, and we can assemble all the phenomena and obtain a more complete result with spontaneous spiral emergence, streaming and aggregation. This is shown in fig. 5. Here the amoebae distribution and c_0 are the same as in fig. 3. Now, it was possible to remove the condition of a movement constrained to the amoebae being on state 1, as well as using a constant acceleration a in the $\overrightarrow{x}$ and $\overrightarrow{y}$ for the movement of the amoebae. Supplementary video 3 allows to observe the same simulation with more detail.

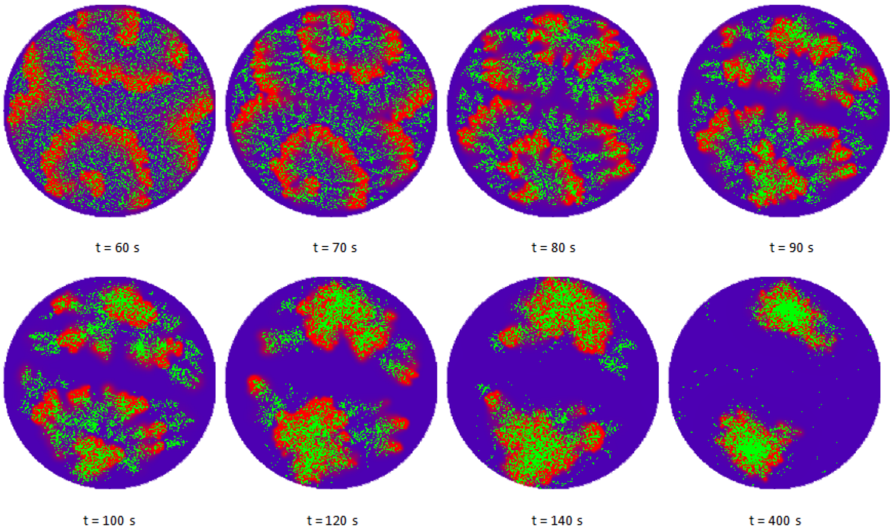


Figure 5: Observations of streaming and aggregation together with self-sustained spiral waves, amoebae move freely with $a = 2\ nm.s^{-2}$. The other simulation parameters are $c_0 = 0.2\ nmol.mm^{-2}$, $\tau = 0.4\ s$, $t_r = 10\ s$, $\Gamma = 0.5\ s^{-1}$, $c_T = 0.5\ nmol.mm^{-2}$ and $\Delta c = 300\ nmol.mm^{-2}$.

3.4 Directional sensing effect

We now proceed to analyze the effect of directional sensing in the model. We test this effect for high and low frequencies, with the constraint of amoebae only moving when they are on state 1, as in section 3.2, and for free motion in the conditions of the two established spirals of section 3.3. Fig. 6 allows to observe that the directional sensing effect is in fact a feature required to complete the aggregation in this model. Here it can be seen that for low frequencies its effect is more pronounced (whether in free or constrained motion, responding to a pacemaker or to spontaneously formed spirals), whilst for higher frequencies it doesn't have an effect and the pattern formed with and without directional sensing is the same.

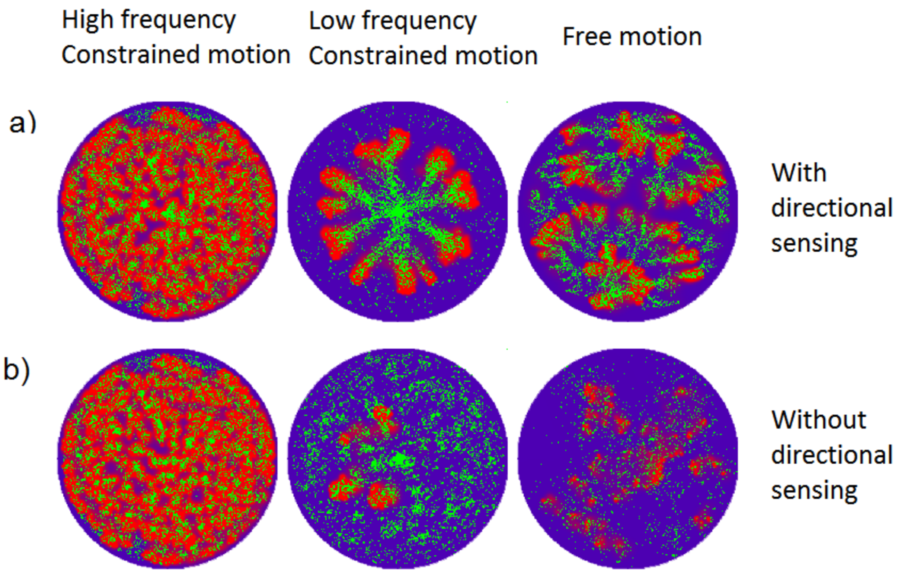


Figure 6: The system configuration for the time instant $t = 50\ s$. A) with directional sensing. B) without directional sensing.

3.5 Self-Organising complex regular patterns

We will now characterize the free-motion adaption made on the model, by removing the condition of amoebae only moving when on state 1, i.e., when emitting cAMP. We put again the pacemaker in the middle of the circular domain and allow the amoebae to move with constant velocity in direction of the gradient, with directional sensing.

Bifurcating parameters were found to be degradation rate Γ and amoebae velocity v. For a given set of parameters, a surprising outcome was found: the emergence of symmetric self-organised regular steady state patterns. Fig. 7 shows three of these symmetries.

The emergent symmetries here obtained consist either of a $90°$ symmetry or a mirror symmetry. Supplementary video 4 allows to see the formation of another one of these patterns for parameters $\Gamma = 0.3\ s^{-1}$ and $v = 480\ nm.s^{-1}$.

Symmetries and patterns are often present in nature, being the snowflakes and hexagonal beehives some of the most emblematic cases. These symmetries

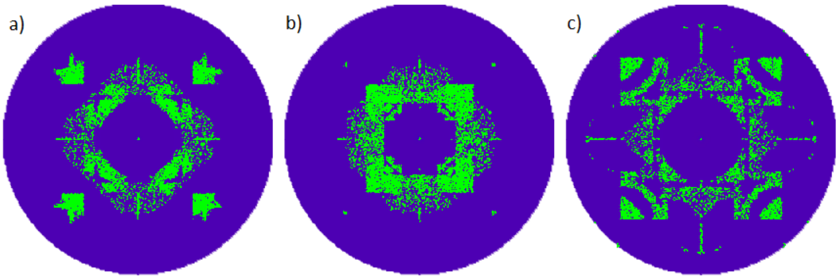


Figure 7: Emergent regular amoebae steady state patterns. After a 90° rotation, the pattern remains invariant. a) $v = 320\ nm.s^{-1}$, $\Gamma = 0.3\ s^{-1}$. b) $v = 320\ nm.s^{-1}$, $\Gamma = 0.5\ s^{-1}$. c) $v = 480\ nm.s^{-1}$, $\Gamma = 0.25\ s^{-1}$. All with $t_R = 20\ s$, $\tau = 0.2\ s$, $\Delta c = 150\ nmol.mm^{-2}$, $c_T = 0.5\ nmol.mm^{-2}$.

consist on an interesting result of our model, for they are of considerable complexity and emerge naturally for a given parameter zone. It is verified that, fixing the bifurcating parameters, any initial amoebae distribution will give rise to the same symmetry pattern.

4 Conclusions

We conclude a thorough analysis of a simple discrete model constructed based on features of two previous models and adapted iteratively to better reproduce the reality of *Dictyostelium* aggregation.

The model is successful in reproducing spiral wave formation, streaming, aggregation in the diffusive center and highlights the direct influence of parameters such as amoebae velocity, phosphodiesterase activity and period wave emission on the dynamics of the system. Its simplicity in not including any biochemical derived term or equation allowed us to interpret the main phenomena of the system as emerging from the non-linear dynamics of reaction-diffusion systems in excitable media.

In this study, it was possible to exhibit the essential features of *Dictyostelium* aggregation and thus better understand the nature of pattern formation in these starving colonies. Despite the high number of parameters in this model and the assumption of motion of the amoebae with constant velocity, the calibration of this type of models will enable the determination of some of the physical parameters of the system. On the other hand, the new type of regular pattern found here presents a challenge in laboratory observations.

References

[1] Kessler D.A. and Levine H., (1993) Pattern formation in *Dictyostelium* via de dynamics of cooperative biological entities. *Phys. Rev. E* **48** (6): 4801-4804.

[2] Dilão R. and Hauser M.J.B., (2013) Chemotaxis with directional sensing during *Dictyostelium* aggregation. *Comptes Rendus Biologies* **336**: 565-571.

[3] Dilão R. and Sainhas J., (1998) Validation and calibration of models for reaction-diffusion systems. *Int. J. Bif. and Chaos* **8**(6): 1163-1182.

[4] Weijer C.J., (2004) Dictyostelium morphogenesis, (review article). *Curr. Opin. Genet. Dev.*, **14**(4):392-398.

[5] Kessin R.H., (2001) Dictyostelium, Evolution, Cell Biology and the Development of Multicellularity. *Cambridge University Press*, 1st edition.

[6] Martiel J.L. and Goldbeter A., (1987) A model based on receptor desensitization for cyclic AMP signaling in Dictyostelium cells. *Biosphys. J.*, **52**:807-828.

[7] Levine H., Aranson I., Tsimring L. and Truong T.V., (1996) Positive genetic feedback governs cAMP spiral wave formation in *Dictyostelium*. *Proc. Natl. Acad. Sci.*, **93**:6382-6386.

[8] Vasiev B. N., Hogeweg P. and Panlov A. V., (1994) Simulation of *Dictyostelium discoideum* aggregation via Reaction-Diffusion Model. *Phys. Rev. Lett.*, **73**(23):3173-3176.

[9] Palsson E. and Cox E.C., (1996) Origin and Evolution of circular waves and spirals in *Dictyostelium discoideum* territories. *Proc. Nat. Acad. Sci.*, **93**:1151-1155.

[10] Lauzeral J., Halloy J. and Goldbeter A., (1997) Desynchronization of cells on the developmental path triggers the formation of spiral waves of cAMP during *Dictyostelium* aggregation. *Proc. Nat. Acad. Sci.*, **94**:9153-9158.

[11] Tyson J.J., Alexander, K.A., Manoranjan V.S., and Murray J.D., (1989) Spiral waves of cyclic AMP in a model of slime mold aggregation. Physica A, **34**:193-207.

[12] Dallon J.C., B. Dalton, and C. Malan, (2011) Understanding streaming in *Dictyostelium discoideum*: theory versus experiments. *Bull. Math. Biol.*, **73**(7):1603-1626.

[13] Dallon J., Jang W., and Gomer R.H., (2006) Mathematically modelling the effects of counting factor in *Dictyostelium discoideum*. *Math. Med. Biol.*, **23**:45-62.

[14] Nagano S., (1998) Diffusion-assisted aggregation and synchronization in *Dictyostelium discoideum*. *Phys. Rev. Lett.*, **80**(21):4826-4829.

[15] Kriebel P. W., Barr V.A, and Parent C.A., (2003) Adenyl Cyclase localization regulates streaming during chemotaxis. *Cell*, **112**:549-560.

[16] Hilgardt C., Cejková J., Hauser M.J.B., H. Seveikova H., (2008) Streamless aggregation of *Dictyostelium* in the presence of isopropylidenadenosin. *Biophysical Chem.*, **132**:9-17.

In virtuo modelling for biologists: How to design graphical interface for the RéISCOP simulation platform?

Guillaume Longelin Péron[1], Gireg Desmeulles[1]

[1]Lab-STICC, UMR 6285 CNRS, UEB/ENIB/CERV, France

Abstract

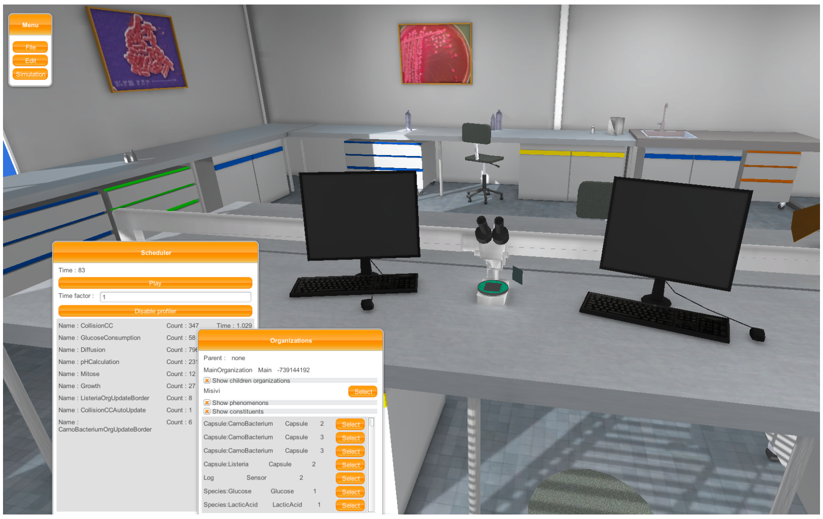
This article presents a future component of the RéISCOP simulation platform. RéISCOP provides an interaction-based meta-model to build models for simulations. Currently, model construction needs to be done by a programmer. This is a problem for the biologist who cannot construct models by himself. We propose to include a graphical interactive interface to allow biologists to build models in RéISCOP with help of *in virtuo* experiments.

1 Introduction

In 2013, the Royal Swedish Academy of Sciences decided to award the Nobel Prize in Chemistry to Martin Karplus, Michael Levitt and Arieh Warshel for the development of multiscale models for complex chemical systems. These works stress the point that mathematical modelling and computer simulation of complex phenomena are more and more central to the field of fundamental research applied to living systems.

In this context, Virtual Reality (VR) may become essential to study complex systems such as biological systems. VR places the user at the heart of a virtual laboratory, so that he can use tools which share similarities with experimental science methods: the user (*i.e.* the biologist) can therefore investigate the virtual biological world using various methods such as 3D visualisation and interactions, numerical methods, etc. We usually call this kind of investigations "*in virtuo* experiments" for its similarities with the expressions *in vivo* and *in vitro* [1]. *In virtuo* experiment is a subset of computer simulations studies called *in silico*. It replaces user on experiment context by immerse him in simulation environment, for example a biology laboratory (Figure 1). He can use VR elements to interact with environment. This places *in virtuo* experiments at the intersection of biology and VR. Humans are then directly involved in the *in virtuo* experimentations of the numerical models within the virtual environment. RéISCOP is both, a framework and a meta model that have been designed to enable the *in virtuo* experiments. Recently, it has been rewritten into a version 2.0 that includes meta-model, interactive simulation engine and reaction/diffusion and bacteria models. The GUI (Graphical User

Interface) that is necessary to enable the *in virtuo* experiments has not been developed yet. This position paper addresses our work on the design and implementation of an efficient graphical user interface. First, we present RéISCOP 2.0. Then we review the various multi-agent platforms by focusing on the proposed interfaces; Finally we conclude about the scientific challenges that we must overcome to imagine an relevant graphical user interface for *in virtuo* experience.



Time : 83

Time factor : 1

Name : CollisionCC	Count : 347	Time : 1.029
Name : GlucoseConsumption	Count : 58	
Name : Diffusion	Count : 79	
Name : pHCalculation	Count : 23	
Name : Mitose	Count : 12	
Name : Growth	Count : 27	
Name : ListeriaOrgUpdateBorder	Count : 8	
Name : CollisionCCAutoUpdate	Count : 1	
Name : CarnoBacteriumOrgUpdateBorder	Count : 6	

Parent : none

MainOrganization Main -739144192

☒ Show children organizations

Misivi

☒ Show phenomenons

☒ Show constituents

Capsule:CarnoBacterium	Capsule	2	Select
Capsule:CarnoBacterium	Capsule	3	Select
Capsule:CarnoBacterium	Capsule	3	Select
Capsule:Listeria	Capsule	2	Select
Log	Sensor	2	Select
Species:Glucose	Glucose	1	Select
Species:LacticAcid	LacticAcid	1	Select

Figure 1: Virtual biology laboratory on RéICOP simulation. Bacteria development simulation take place on this environment. We can see it by zooming in on microscope (Figure 6)

2 RéISCOP 2.0

RéISCOP is a meta-model and a C# written simulation platform using Unity3D as basement for graphical user interface. Its name is an acronym of following sentence that gets main concepts of meta-model together – **Re**ification of **I**nteractions, **S**ystems, **C**onstituents, **O**rganizations and **P**henomenons. We can refer to [2] for a description of the former 1.0 version to understand all meta-model notions. The 2.0 version has not been published yet but we can present some points[1].

[1] Documentation can be found at: http://www.cerv.fr/ReISCOP/doxygene/index.html

2.1 Meta-model

The RéSCOP meta-model provides a means to create an interaction-based simulation that can be seen as a dual method for individual based model (IBM) simulation (Figure 2).

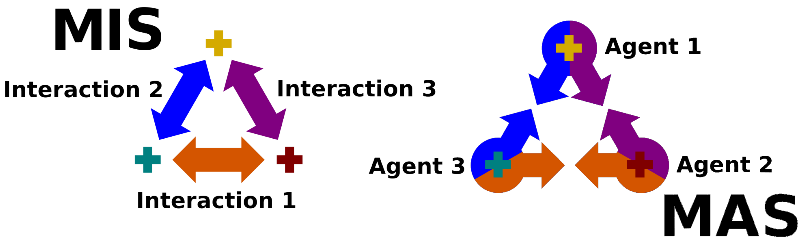


Figure 2: In multi-agent systems (MAS), individuals are explicit and interactions implicit whereas in multi-interaction systems (MIS), interactions are reified and individuals become implicit.

The meta-model can be used to implement models or simply to discuss and to design on paper. We briefly describe here the main concepts of the modelling formalism:

Constituent: *Constituents* are variables, parameters or quantities involved in the models. At each moment, they represent the current state of the model. *Constituent* may model a concentration, a 3D shape, a position or a diffusion coefficient. In our graphical notation, constituent is represented by ”+”.

Interaction: *Interactions* are the processes that modify the states of the *constituents* over time. An *interaction* points towards a constant set of *constituents* with read or write access, at every step of simulation. *Interaction* may model a chemical reaction, a mechanical collision, a chemical diffusion. Interaction are represented by a multi-head arrow. Each arrow head points out a constituant.

Phenomenon: *Phenomena* are in charge of producing *interactions* during the simulation. A *phenomenon* focuses on the states of *constituents* (only those which are known by its *organization*). If it detects that the required conditions are satisfied, then it produces a new *interaction*. In this way, depending on its type, each *interaction* belongs to a *phenomenon*. Link between a phenomenon and its *interactions* is represented by a dotted line.

Organization: An *organization* represents the dynamics of a system part. It is composed of *phenomena* which themselves are composed of *interactions*.

In addition, the *organization* handles a set of *constituents* which represent the static part of the system. The *organization* role is to maintain the consistency of that set of *constituents* over time. Finally, an *organization* can be composed of sub-*organizations* corresponding to sub-systems.

System: *Constituents, interactions, phenomena* and *organizations* are used to model and populate our virtual worlds with autonomous *systems* (Figure 3).

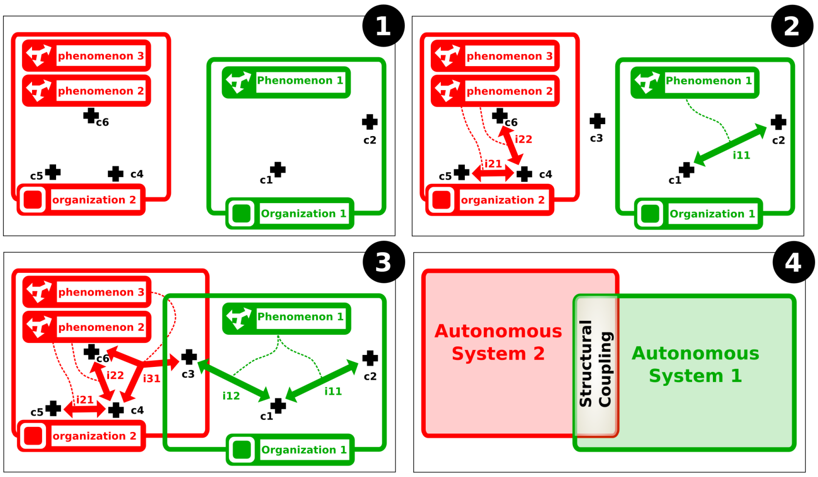


Figure 3: RéISCOP dynamics. 1: Organizations 1 and 2 have phenomena and know constituents. 2: If conditions are satisfied, phenomena instanciate interactions; a new constituent (c3) is created. 3: Organizations adapt their boundaries according to their internal rules; phenomena detect new conditions for creating interactions; new interactions are created. 4: We have here two autonomous systems coupled by a part of their structure. A concrete model can be seen on figure 5

Constituents represent the static part of *systems*. We call *structure* the whole set of *constituents*. *Interactions, phenomenon and organizations* define the dynamics. By doing so, we focus on the dynamics rather than on the statics. We organise the dynamics instead of structuring the *system* state, assuming that the method is less reductionist and more suited to the study of complex *systems*. Furthermore, multi-interaction *systems* are connected by structural coupling [3] (Figure 4). This differs from the usual component based approach which uses input/output connections between components. Here, *systems* perceive one another through perturbations on their own structure. The autonomous nature of *systems* is thus consolidated.

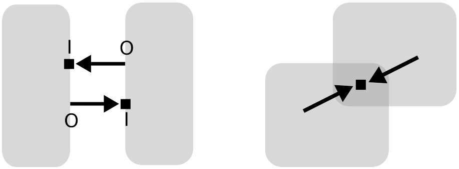


Figure 4: Classical input/output interface between two systems (left) and structural coupling between two systems (right)

2.2 Platform

RéISCOP software implements the meta model and can build and simulate models (Figure 5) described with XML files and C# delegate functions. Currently, simulation engine work properly (Figure 6). We use it to simulate bacterium colonies development in the context food security project however it is not an "*in virtuo*" experimentation. Indeed, Models have to be built by writing XML description file. Model editor is missing from 3D RéISCOP simulator. RéISCOP 1.0 had two distinct interfaces: a 3D simulator and a 2D editor. The solution will be to include model building on simulator. For this, we need to study graphical user interface of actual multi-agent platforms.

3 Agent-based simulation platform

We have reviewed actual multi-agent simulation platforms to study their graphical user interfaces for models construction. We have reviewed this platforms because their models paradigms (multi-agent system) are close to RéISCOP meta-model paradigm (multi-interaction system). We focus on three factors:
Simplicity: How easy is it to build model? Can a biologist build model alone?
Expressiveness: Is it possible to build complex model?
Interactivity: Do model and simulation are easy to manipulate?
This review shows us different examples of agent-based simulation building. We have examined three platforms that are frequently used for simulation modelling with multi-agent systems: Repast, NetLogo and GAMA; and finally, two platforms that use graphical elements to construct model for simulation: AgentSheet, SeSam.

Repast: Repast is a Java framework for agent-based simulation[5]. It allows the creation of an agent-base simulation using the Java and it includes a library of object to create, run, display and collect data from agent-based simulation. Repast models may only be build with Java programming language. This is an obstacle for those who want to build their models alone but who are unable to program (as is the case for many biologists). Although Repast framework

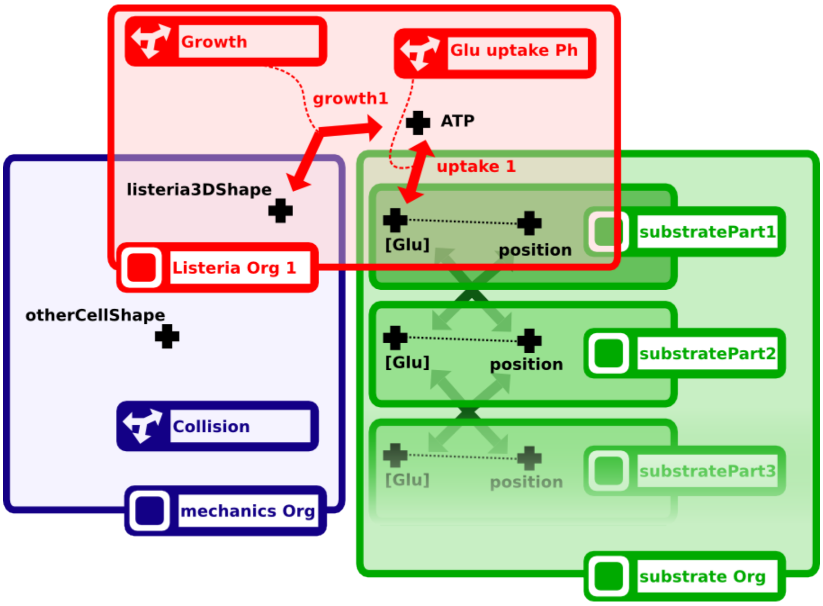
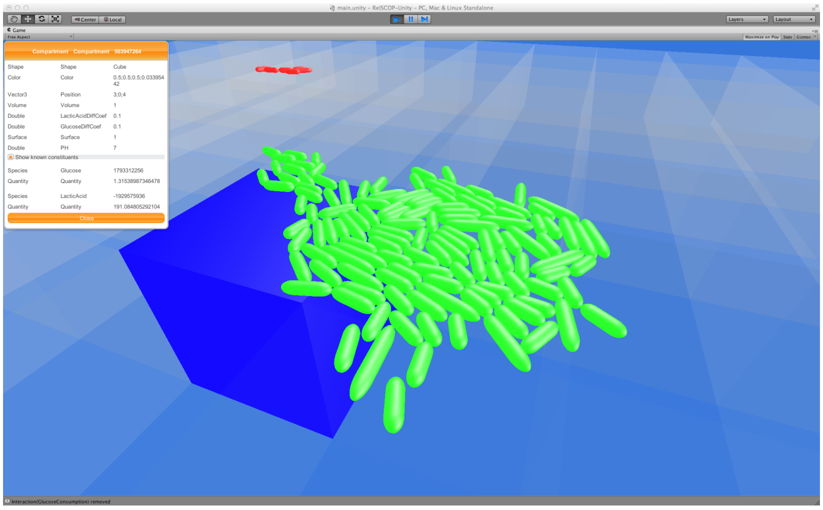


Figure 5: This figure shows a RéISCOP model containing 3 sub-systems : (red) a Listeria that is coupled with (green) a part of discretized chemical substrate and (bleu) a mechanical organization handling the possible interactions between the different cell shapes. For example, *Listeria Org 1* is an organization that is responsible for the cell is coupled with correct constituent *Glu* and cell *position* correspond with its shape location. If it is necessary, *Collision* phenomenon instantiate collision interactions between shapes. *Glu* constituent is a real number that represents glucose concentration in a mesh of discretized substrate. Every simulation step, *growth1* interaction decreases *ATP* value and increases shape size.

allow to write varied and complex model with the help of Java programming language and framework tools.

Netlogo: NetLogo is a multi-agent programming language and modelling environment [6]. This platform is designed for research and education for a large range of disciplines. The NetLogo language is a logo language variation. With this language, *turtles* represent agents during the simulation. They are located agents that move on spacial agents: *patches*. It hopes teach multi-agent simulation programming. Build a model on NetLogo is more simple than on Repast but models are more limited. The platform uses a programming language that may be a problem to model building for non-programmer. The platform offers parametric simulation interface to edit simulation parameters.

GAMA: GAMA is a modelling and simulation environment for building spatially agent-based simulation [7]. It is originally used for simulate geographic



Compartment Compartment 503947264

Shape	Shape	Cube
Color	Color	0.5;0.5;0.5;0.03395442
Vector3	Position	3;0;4
Volume	Volume	1
Double	LacticAcidDiffCoef	0.1
Double	GlucoseDiffCoef	0.1
Surface	Surface	1
Double	PH	7

☒ Show known constituents

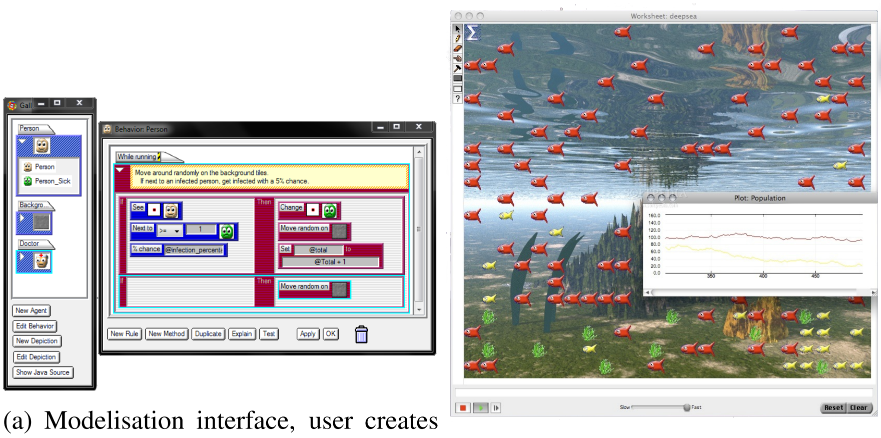
Species	Glucose	1793312256
Quantity	Quantity	1.31538987346478
Species	LacticAcid	-1929575936
Quantity	Quantity	191.084805292104

Interaction(GlucoseConsumption) removed

Figure 6: RéISCOP simulation interface. We see two bacteria colonies: green bacteria in foreground are listeria and red bacteria in background are carnobacterium. Bacterial colonies live on an agar-based growth medium on Petri dish.)

information system but must be extend to other domains. GAMA makes available an agent-oriented language GAML (GAma modelling Language). It is more simple to build a model with this language than Java because it is structured with multi-agent system elements like agents, agent actions, agent reflex, etc. GAMA supplies a graphical editor to simply build simulation with minimum code writing. But this interface is only used during first part of a project to structure simulation elements (agents, theirs actions, theirs reflexes) after this, it must be exported to GAML and user writes agent actions code. GAMA is easier to use than Repast because it has its own, specifically developed, simulation language. But platform has important coding part just as Repast and NetLogo.

AgentSheets: AgentSheets is a platform that provides graphical tools to construct simulation model [8]. Non-programmer can simply use AgentSheets to build model. AgentSheets consists of two main interfaces, a interface to build models and an other to launch simulations. In modelling interface, user can add new agents and specify their behaviours. Behaviours are formed by condition-action statements (Figure 7a). In simulation interface, user can instantiate agent in environment called *agentsheets* (Figure 7b). An AgentSheets function allows user to select an agent in simulation interface and see verified conditions and realised actions in modelling interface. This functionality can help user to understand relationship between an agent in modelling interface and agents



Move around randomly on the background tiles.
If next to an infected person, get infected with a 5% chance.

(a) Modelisation interface, user creates agents and specifies theirs behaviors with condition/action components.

(b) Simulation interface, agents can be add in environment.

Figure 7: AgentSheets interfaces

with the same behaviour in simulation interface. This platform is limited for complex system building. Simulations look like cellular automaton. This simulation platform can not be used in biologic simulation because of limited possibilities.

SeSam: SeSam is a platform for modelling and experiment with agent-based simulation [9]. SeSam allows to build agent behaviour with UML-like activity diagrams. Next, user can use a menu to select actions that will be realised during states and conditions between states. User have functions list that can be used for an action. He can specify function parameters with other functions and so on. Edition menus are not user friendly and it is difficult with first look to understand action building. An other difficulty is to edit existing models.

On one hand, there are three first platforms frequently use to build multi-agent simulations that are expressive enough to be used for biological simulation. But building models with this platforms is not easy for non-programmers. On the other hand, two other platforms allow user to construct model with graphical tools. First, SeSAm simplifies organization agent behaviours but edit menu makes action specifications harder. Next, AgentSheets makes successfully easier model building with graphical elements. Any platform has a clear and interactive model representation and allows to create complex model (Figure 8).

	Repast	NetLogo	GAMA	AgentSheets	SeSam
Simplicity	- - -	- -	- -	++	-
Expressiveness	++	+	++	- -	+
Interactivity	- -	-	-	+	+

Figure 8: Summary of platforms studied factors.

4 Graphical user interface to design dynamical system

We have not found an existing method to manipulate complex meta-model with graphical interface. There is only AgentSheets that includes a helpful but limited graphical user interface to build model. At present, usage of RéISCOP meta-model graphical representation is only theoretical. We already use it to describe model with XML files, but we would like to employ it through *in virtuo* experiments. RéISCOP meta model implements concepts like object and class although a biologist does not know oriented object concepts. We would like to clarify concept of instantiation for non-programmer. For example, a phenomenon can apply a physical collision with instantiation of interaction between two constituents. Currently, static model handle dynamic system design. We want to give the possibility to construct model dynamically. We would like to provide non-programmers with Unity3d modelling interface that will be included in RéISCOP simulation platform.

5 Conclusion

Our objective is to include a graphical interface for model building in RéISCOP simulator. User must easily interact with it. Concepts of abstraction need to be clearly understood by model builder. Reviewed platforms do not give us tools to manipulate models dynamically. That's why we want add to RéISCOP an interactive models building interface. We need to create interaction metaphors to build models. This metaphors could be clear for non-programmers. But we have to keep a balance between simplicity to build models and complexity of made models. Model and simulation representations are not in same graphical and concept spaces. The main challenge will be that simulations and constructions spaces work together. In order to do this, a method will be to display graphical connections between elements on simulation interface and items on modelling interface.

In the *in virtuo* experiment context, we plan to interface RéISCOP platform with virtual reality equipment to increase user immersion. It might help to design models.

References

[1] J. Tisseau, "Réalité virtuelle: autonomie in virtuo," *Habilitation á Diriger des Recherches*, 2001.

[2] G. Desmeulles, S. Bonneaud, P. Redou, V. Rodin, and J. Tisseau, "In virtuo experiments based on the multi-interaction system framework: The réiscop meta-model," *Computer Modeling in Engineering and Sciences (CMES)*, vol. 47, no. 3, pp. 299–329, 2009.

[3] P. Dumouchel, P. Bourgine, and F. J. VARELA, *Autonomie et connaissance*. Seuil, 1989.

[4] S. F. Railsback, S. L. Lytinen, and S. K. Jackson, "Agent-based simulation platforms: Review and development recommendations," *Simulation*, vol. 82, pp. 609–623, 2006.

[5] N. Collier, "Repast : An extensible framework for agent simulation," *The University of Chicago's Social Science Research*, vol. 36, pp. 371–375, 2003.

[6] S. Tisue and U. Wilensky, "Netlogo: A simple environment for modeling complexity," in *International Conference on Complex Systems*, pp. 16–21, 2004.

[7] A. Grignard, P. Taillandier, B. Gaudou, D. A. Vo, Q. Huynh, and A. Drogoul, "Gama 1 . 6 : Advancing the art of complex agent-based modeling and simulation," in *PRIMA 2013: Principles and Practice of Multi-Agent Systems*, pp. 117–131, 2014.

[8] A. Repenning and T. Sumner, "Agentsheets: A medium for creating domain-oriented visual languages," *Computer*, vol. 28, no. March, pp. 17–25, 1995.

[9] F. Klügl, R. Herrler, and C. Oechslein, "From simulated to real environments: How to use sesam for software development," in *Multiagent System Technologies*, pp. 13–24, 2003.

Modeling of intercellular transport for emerging applications in synthetic biology

Loïc Talide[1], Zoé Blanck[1], Marine Renou[2], Thibault Wallois[2], Elise Rosati[2,3], Morgan Madec[2,3], Christophe Lallement[2,3], Jacques Haiech[4]

[1] Ecole supérieure de Biotechnologie de Strasbourg (ESBS), 300 Boulevard Sébastien Brandt, F-67412 Illkirch Cedex 02

[2] Télécom Physique Strasbourg (TPS), 300 Boulevard Sébastien Brandt, F- 67412 Illkirch Cedex 02

[3] Laboratoire des Sciences de l'Ingénieur, de l'Informatique et de l'Imagerie (ICube), UMR 7357, Equipe SMH, 300 Boulevard Sébastien Brandt, F-67412 Illkirch Cedex 02.

[4] Laboratoire d'Innovation Thérapeutique (LIT), UMR 7200, 74 route du Rhin, F-67400 Illkirch.

Abstract

Synthetic biology is a way to create new biological functions that do not exist in nature to meet specific needs (*e.g.* targeted drugs, diagnostic microsystems for healthcare applications, bio-fuels in the field of the environment ...). Nowadays, artificial bio-functions become more and more complex. They use different biochemical mechanisms to achieve the targeted function. One of the most common consists in designing new gene regulatory networks. Nevertheless, one of the main bottlenecks is the integration of a large number of artificial genes in the host cell. A promising way to get around consists in dispatching the function in multiple host cells and makes them work together in a kind of micro-ecosystem. *In silico* design of such systems requires predictive models of intercellular transport of molecules. This issue has been tackled through two projects carried out by master students from different background (biotechnologies for some of them, microelectronics and computer science for the others). An overview of intercellular transport modeling is given in the first part of this paper. Then, models are illustrated in two examples developed during student projects.

1 Emergence of multi-cellular systems in synthetic biology

Synthetic biology can be defined as the application of engineering principles to the fundamental components of biology. In particular, the design of artificial gene regulatory networks is one of the most investigated way to design new biological functions. Although very promising, this technology suffers from two main drawbacks that may limit the complexity of the artificial function. First, the number of artificial genes that can be added to given microorganism is generally quite small (some units). Second, artificial genes designed for the application should be independent with each other and with the genome of the host cell, *i.e.* any potential interaction (cross regulation) between the artificial and the rest of the genome should be avoided. A way to overcome these drawbacks is to split the main function and to implement each sub-function into different host cells [1]. This way, some components (regulating proteins, promoters ...) may be used several times inside different host cells. However, sub-functions are not completely independent; the signal transfer mechanisms between cells must also be designed [2].

Examples of systems made with several reprogrammed bacteria achieving a complex function have already been developed (digital gates [3], prey-predators ecosystems [4] and programmable pattern generator [5]). For such systems, the proof of concept is carried out through in silico simulations but most of the time, models are simplified and the intercellular transport of molecular species is roughly described. Assumptions used are valid for small systems but may lead to inaccuracy for complex ones. This paper deals with the design-oriented modeling of such mechanisms. The first section is an overview of the intercellular transport mechanisms and the associated models. Then, two examples are given.

2 State-of-the-art in intercellular transport modeling

Exchange of chemical species between cells involves many steps and these steps are localized in space: the molecule have to move inside the sender organism to the plasma membrane, cross this membrane, diffuse through the intercellular medium, cross again the plasma membrane of the receptor and move inside the receptor cell to the place where it may have an activity. All these movements could be modeled thanks to four different mechanisms: a simple random diffusion, a passive transport, an active transport and exo/endocytosis. Let us first have an overview of these mechanisms and the associated models.

The **random diffusion** corresponds to the motion of chemical species inside cells or through the extracellular medium. Except for very specific application, the intracellular motion is ignored (the concentration of the species near the membrane is equal to the mean concentration inside the cell). Conversely, extracellular motion needs to be taken into account. There are several ways to model 2-D or 3-D particle motion [6]. To save simulation time, a compartmental model is often implemented, at least during the early stages of the design process. The equation which describes the diffusion of molecules from a point A to a point B in an unbounded plane is:

$$\frac{d[X_B]}{dt} = \frac{\gamma}{d_{AB}} \cdot ([X_B] - [X_A]) - D \cdot [X_A]$$

where $[X_A]$ and $[X_B]$ are the concentration of X respectively at the point A and B, d_{AB} is the distance between A and B, γ is the diffusion constant (in $\mu M \cdot s^{-1}$) and D is a decay constant modeling the probability that the species is degraded before reaching B.

The passive transport could be compared to simple diffusion through the plasma membrane. As for other diffusion problems in physics, the transport rate is directly proportional to the gradient of the concentration between both sides of the membrane. It does not consume energy. The differential equation that governs this passive transport is:

$$\frac{d[X_{out}]}{dt} = -\frac{d[X_{in}]}{dt} = \alpha \cdot S \cdot ([X_{out}] - [X_{in}])$$

where $[X_{out}]$ and $[X_{in}]$ are the concentration inside and outside the cell, α is the surface permeability coefficient (in $s^{-1} \cdot \mu m^{-2}$) which depends on physic-chemical parameters of the membrane and S is the membrane surface.

The active transport of molecules through the membrane requires two elements: energy (ATP in biology) and a specific transporter integrated in the plasma membrane. This transporter binds ATP, which creates a species flow that occurs in the direction forward or inverse their concentration gradient. Every time an ATP binds a transporter, it is hydrolyzed and the transporter is recycled. Thus, to simplify the model, the transporter is never consumed by the transport mechanism. Sometimes, the transport occurs only when the concentration of the species to transport is above a given threshold Xth. The model is the following (ATP is in excess so its concentration is not involved in the equations of the model):

$$\frac{d[X_{out}]}{dt} = -\frac{d[X_{in}]}{dt} = \begin{cases} 0, & if [X_{in}] < Xth \\ V_{max} \cdot \frac{[X_{in}]}{[X_{in}]+k_P} - \frac{Xth}{Xth+k_P} & otherwise. \end{cases}$$

where V_{max} is the maximum transport rate (in s^{-1}), $[Y]$ is the concentration of the transporter and k_P is a dissociation constant. In order to avoid discontinuities, this equation may be replaced by a standard smoothing function as following:

$$\frac{d[X_{out}]}{dt} = \frac{1}{2}\left(f([Y],[X_{in}]) + \sqrt{\epsilon^2 + f^2([Y],[X_{in}])}\right)$$

with

$$f([Y],[X_{in}]) = V_{max} \cdot [Y] \cdot \frac{[X_{in}]}{[X_{in}] + k_P} - \frac{Xth}{Xth + k_P}$$

and ϵ corresponds to the transport rate for $X = Xth$. It should be noticed that active transport may occur in both direction (toward or from the cell) but each direction requires a specific transporter. The model is the same and the sign of the constant V_{max} corresponds to the transport direction.

The **exocytosis** is the third kind of molecules transport. This way of motion requires energy and the formation of vesicle that will fuse to the membrane. Exocytosis is very important in the brain for the motion of the neurotransmitter through the synapse. The modeling of the exocytosis is very tricky because of the large number of factors involved [7] and will not be discussed in this paper.

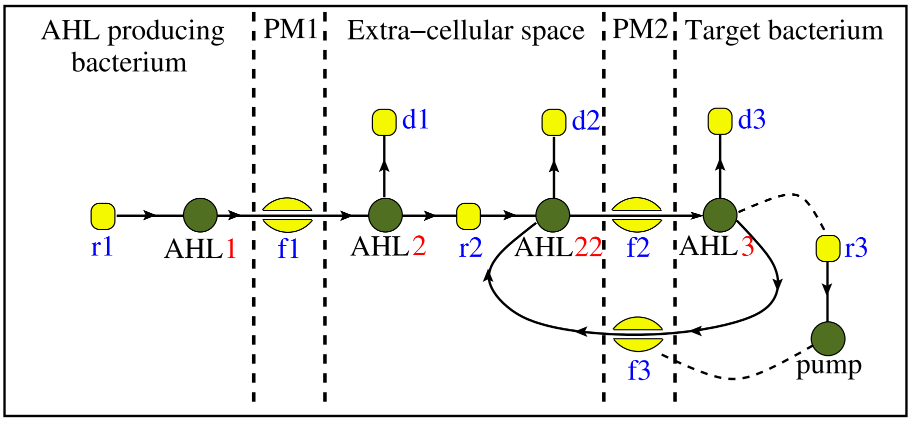


Figure 1: The Virtual Cell cartoon which corresponds to the Example #1. AHLs are produced inside the first bacteria and freely diffuse through its membrane and the extracellular medium to the second bacteria. Two transports are in competition at this point: a passive transport through which AHLs may go inside the receiver and an active transport (efflux pump) that send back AHLs in the extracellular medium. The active transport is controlled by the concentration of pump which is itself synthesized by a gene activated by the AHL. AHL1, AHL2, AHL22 and AHL3 are respectively the concentration of AHLs inside the sender, in the extracellular medium near to the sender, in the extracellular medium near to the receiver and inside the receiver. *r1*, *r2* and *r3* are three reaction rates which correspond respectively to the synthesis of AHL in the sender, a model of the diffusion of AHL in the inter-cellular area and the synthesis of the pump controlled by AHL rate. *f1*, *f2* and *f3* are respectively the flux of AHL through the membrane of the sender and through the membrane of the receiver (one flux corresponding to the passive transport and one to the reverse active transport). Finally *d1*, *d2* and *d3* models the decay of AHL in the three compartments.

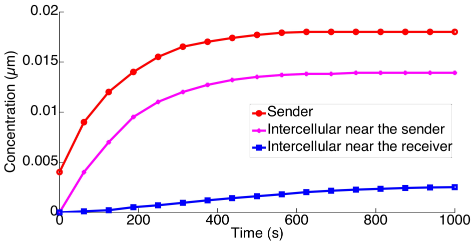
3 Example #1: Modeling of a transport mechanism with a competition between passive and active transport

In order to illustrate our purpose, we model a simple system which consists in a source of N-Acyl Homoserine Lactone (AHL), a hormone widely used in synthetic biology because of its natural quorum sensing and communication roles in bacteria [8], and a receiver cell. The system is designed to be implemented in *E. coli* bacteria. It has been first described and simulated with The Virtual Cell (Fig. 1).

In more details, the model consists in 7 equations rate: (i) the constitutive synthesis of AHL in the production bacteria, (ii) the passive transport through the membrane of the production bacteria, (iii) the random diffusion of AHL between the two bacteria (including decays), (iv) the passive diffusion through the sender membrane, (v) the decay of AHL inside the sender, (vi) the synthesis of the pump which is activated by AHL and (vii) the active transport through the sender membrane which is controlled by the concentration of pumps. The equations used correspond to the ones described in previous section.

The simulation parameters are the following: AHL production rate is set to $0.05\ nM \cdot min^{-1}$ and its decay rate to $1.8 \cdot 10^{-3} min^{-1}$. The surface permeability coefficient of the sender membrane is $0.5\ min^{-1} \cdot \mu m^{-2}$ and the surface of the E.coli is estimated to about $4.83\ \mu m^2$. The diffusion in the extracellular medium is set to $5\ \mu m \cdot min^{-1}$ and the distance between the sender and the receiver to $10\ mm$. For the active transport, the maximal transport rate is $4000 nM \cdot min^{-1}$ and the dissociation constant is $0.25 min^{-1}$. The complete Virtual Cell model can be found in the public BioModels library (talide: assb_blanck_talide).

Simulation results are given in Fig. 2. The concentration of AHL in the sender, the receiver and the intercellular medium is monitored. As expected, a small decay is observed between AHL1 and AHL2 due to the degradation and the dilution that occurs in both intra and extra-cellular compartment. The shape of the AHL22 concentration curve suggests a delay between the time when AHLs exit the cell and the time they are near the target bacteria. The difference of steady state between AHL2 and AHL22 corresponds to the random diffusion in extracellular medium. Finally, the level of AHL3 in the target cell depends on the efficiency of the efflux pump (V_{max}) as well as the threshold concentration (X_{th}) beyond which the pump is not active.



(a)

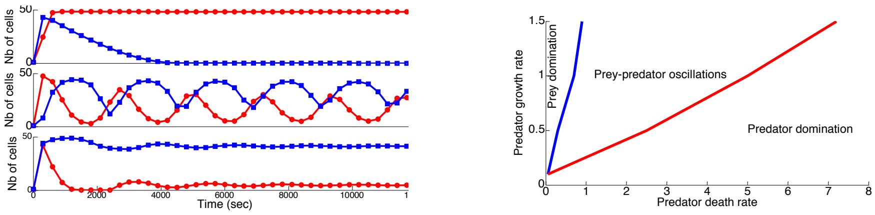
V_{max}	X_{th}	*AHL3*
uM·min^{-1}	***uM***	***uM***
0.4	0	0.1
4	0	0.01
40	0	0.001
4	0.001	0.001
4	0.1	0.005

(b)

Figure 2: Simulation results: (a) transient evolution of AHL concentration as a function of the position (circles, triangles and squares corresponds respectively to the concentration of AHL in the sender, in the inter-cellular area but close to the sender and in the inter-cellular area close to the receiver) and concentration of AHL at the steady state in the receiver as a function of the maximum efflux pump rate and the threshold value.

4 Example #2: Improvement of the model of a prey-predator ecosystem with intercellular transport considerations

The second example concerns the prey-predator ecosystem described in [4]. It is composed of two reprogrammed E.coli strains. The *prey* system consists in constitutive expression of a suicide gene which can be repressed by an antidote activated by the AHL (3O6HSL) synthesized by the *predator*. The *predator* has a non-constitutive suicide gene that requires another AHL (3OC12HSL), synthesized by the *prey*, to be expressed. This double feedback loop leads to equilibrium between the number of *predators* and *preys* that strongly depends on the death and grow rate of both bacteria. Three states may be reached: domination of prey, domination of predators or oscillation. In [4], a rough model of the system is established in order to predict these states through static and transient simulations. The aim of this work is to improve this model in order to take into consideration a passive transport of AHL between both cells. The complete mode which consists in 21 differential equations and 58 parameters is implemented in VHDL-AMS, a hardware description language mostly used in microelectronics domain for the description and the simulation of complex heterogeneous systems. The possibility to efficiently describe biological systems through this language has recently been demonstrated [9]. Simulation results, given in Fig. 3, show that the behavior of the system is in accordance with experimental results and simulations obtained with the rough model and described in [4]. Nevertheless, quantitative results are not exactly the same.



(a) (b)

Figure 3: Simulation results: on the left, transient evolution of the number of prey (red dots) and predators (blue squares) as a function of the growth rate ratio vs death rate of predators. The ratio is equal to 0.7 for the top curve, 4 for the second and 7 for the third. On the right, representation of the border between the three states as a function of the couple (growth rate of predators, death rate of predators).

5 Conclusions and perspectives

This works presented in this paper give an overview of the way intercellular transport of chemical species can be modeled. Although the discussed models are very basic, they are sufficient to take into account the main effect that may occur. The main difficulty encountered in model development is the choice of consistent values for the parameters involved in the transport equations. Indeed, most of them are empirical and do not necessarily correspond to measurable biological signals. They were therefore estimated from values from the literature and / or extracted from experimental results and / or set to an arbitrary value.

The VHDL-AMS implementation performed on the second model is very interesting. Up to now, we demonstrated that the gene regulatory network inside a cell can be modeled by equivalent electronics circuits and, as a consequence, widely analyzed with electronics simulators [9]. According to the equations, transport mechanisms between cells might also been represented by electronic equivalents circuits. As a consequence, this work paves the way to the extension of our modeling formalism to multi-cellular systems.

References

[1] K. Brenner, L. You, F.H. Arnold, “Engineering microbial consortia: a new frontier in synthetic biology”, in Trends Biotechnology, 26(9), 2008.

[2] S. Regot, et al. “Distributed biological computation with multicellular engineered networks”, in Letter of Nature, vol. 469 (2011).

[3] W. Bacchus, M. Fussenegger, “Engineering of synthetic intercellular communication systems”, in Metabolic Engineering, vol. 16 (2013).

[4] F. Balagaddé et al. , “A synthetic Escherichia coli predator–prey ecosystem” in Molecular Systems Biology, 4:187 (2008)

[5] S. Basu et al, “A synthetic multicellular system for programmed pattern formation”, in letter of nature, vol. 434 (2005).

[6] D. Ridgway, G. Broderick, M.J. Ellison, “Accommodating space, time and randomness in network simulation” in Current Opinion in Biotechnology, vol. 17, 2006.

[7] B.A. Earnshaw P.C. Bressloff, “Biophysical model of AMPA receptor trafficking and its regulation during long-term potentiation/long-term depression” in Journal of Neuroscience, vol. 26(47), 2006.

[8] A. Kumari, P. Pasini, S. K. Deo, D. Flomenhoft, H. Shashidhar, and S. Daunert (2006). Biosensing Systems for the Detection of Bacterial Quorum Signaling Molecules. Analytical Chemistry 78, 7603–7609.

[9] Y. Gendrault, M. Madec, C. Lallement, J. Haiech, Modeling biology with HDL languages: a first step toward a Genetic Design Automation tool inspired from microelectronics, IEEE Transactions on Biomedical Engineering, vol. 61, no. 4, pp. 1231–1240, 2014.

A bioinformatics tool to find novel and conserved cell size regulators

Zoltán Dúl[1,2,3], Azeddine Si Ammour[3], N. Shaun[4], B. Thomas[4], Attila Csikász-Nagy[1,2,3]

[1] Randall Division of Cell and Molecular Biophysics, King's College London, London, SE1 UL, United Kingdom

[2] Institute for Mathematical and Molecular Biomedicine, King's College London, SE1 UL, United Kingdom

[3] Research and Innovation Centre, Fondazione Edmund Mach, 38010 San Michele all'Adige, Italy

[4] Department of Haematological Medicine, King's College London, London SE5 9NU, United Kingdom

Abstract

Evolutionally conserved pathways regulate several essential biological processes, such as the cell cycle, DNA replication and protein synthesis. We hypothesize that there exists a conserved regulatory mechanism that controls cell size as well. To test this proposition we developed a bioinformatics tool that collects evolutionarily conserved proteins, which have been described as cell size regulators in genome-wide studies previously. Hence we collected existing large-scale data from five evolutionally distant organisms (*S. cerevisiae, S. pombe, H. sapiens, A. thaliana* and *D. melanogaster*) and looked for conserved orthologous proteins with conserved cell size regulatory functions. This allowed us to identify a core conserved cell size regulatory network and create a list of predicted novel cell size regulators in two of these organisms. Initial analysis found that the key regulator of cell size is the Ribosome Biogenesis pathway, while orthologous proteins of TOR pathway kinases and protein kinase C have affect to the cell size in all five investigated organisms.

1 Introduction

Going through the phylogenetic tree we are experiencing that each organism has its own specific set of genes that lead to specific phenotypes. One can see that a particular gene deletion or duplication among organisms did not fix in the population by accident, rather this has specific evolutional reason [1, 2]. While some of the genes could be highly conserved throughout the phylogenetic tree, others are highly transient or specific to particular species [1]. One of the pressures to select and maintain a particular gene in an organism arises because the gene fulfils a specific function that could be under selection.

These high similarity functional proteins can be divided into two types. Orthologs are genes in different species that evolved from a common ancestral gene by speciation; orthologs retain the same function in the course of evolution [1]. Paralogs are genes related by duplication within a genome and usually evolve new functions [3].

Several published orthological databases exist, which are specialized into two different aspects: the first type clusters pairs of genes with the same biological functions [4, 5, 6], while a second type uses phylogenetic trees to identify functional divergence events [7, 8]. In our research we used databases from the first category.

1.1 Cell size regulation

The size of organisms on this planet is highly variable, as well as the dimensions of cells within an organism. Still, specific cell types maintain their size in a relatively constrained regime. There appear to be genuine size controls such as complex coordination of cell cycle, cellular growth and proliferation in all eukaryotic organisms [9]. These mechanisms, which control cell size and maintain cell size in a particular range have been investigated for several decades [10, 11].

In unicellular organisms the cell size equals the size of the organism; while in multicellular species the combination of the number and the size of its cells determines the size of the organism. Although in advanced multicellular organisms cells form organs and the number of cells first determines the size of the organs, the size of the cells has direct control also on size of organs and tissues [12]. Master hormonal regulators affects general control of tissue and organ sizes, but also affect individual cell sizes [12]. Perturbations of cell size can alter organ size suggesting a relationship between the various level size controls [13, 14].

Maintaining a particular cell size in actively dividing cells needs genuine regulation. The size of organisms generally reflects the balance between growth (blastogenesis) and division [13, 14, 15], while cell division and cell growth maintain the size of individual cells in a particular range [13, 16, 17].

There is a large body of literature focusing on individual cell size regulatory proteins. Quite a few studies dealt with system wide screens on cell size in the most studied organisms [18-24]. These studies used mainly system-wide gene deletion or gene silencing analysis to reveal which genes are important to maintain normal cell size. We use these studies to reach a systems level understanding of the core conserved regulatory network of cell size regulation after investigating the conserved function of orthologous cell size regulator proteins.

Our aim is to determine appropriate functional orthologs of a given protein of a specific function in all five organisms (*A. thaliana, D. melanogaster, S. pombe, S. cerevisiae* and *H. sapiens*). Although currently we are focusing on cell size regulation as a specific function the method can be extended to any other biological function, where genome-wide data is available.

2 Methods and Materials

First we collected the gene deletant or silenced mutants that have cell size related phenotypes (e.g.: smaller, larger) in five species [18-24]. We focused on organisms, which were investigated mainly through system-wide gene knockout or knock-down studies. We collated the data on genes which if perturbed cause a cell size defect in these organisms and built an easily usable and searchable database containing these results.

All gene names were mapped from different organisms specific identifiers to SwissProt UniProt IDs [25]. We used the UniProt.org Mapper tool (`http://www.uniprot.org/uploadlists` [25] May, 2014 version), along with Ensembl Gene database, [26] and BioMart tool (`http://www.ensembl.org/biomart/martview`), hence each of the final entries had only one unique UniProt identifier. Next we created the protein-protein interaction (PPI) network of these cell size regulators and their first neighbours by retrieving the latest version PPIs from the BioGRID database [27] (version 3.2.115) and the IntAct database [28] (version 4.1.4). We added the first neighbours of cell size regulator proteins using Cytoscape network analyser program [29] to extend our analysis to proteins that might not be directly involved in cell size regulation, but closely associated with it.

In the next step we paired each of the proteins along with their selected first neighbours via their UniProt protein identifiers to their orthologs in the other four species by using six orthology databases. These databases were HomoloGene [30], orthoMCL [6], inParanoid [4], eggNOG [5], COG [31] and a manually curated ortholog list by the PomBase curators [32]. Later biological pathway information was added to the protein tables using the KEGG database [33]. For the whole structure of the tool see figure 1.

Finally we combined the databases of the five organisms into an integrated front-end website tool (`http://www.orthologfindertool.com`). You can see on figure 2 and figure 3 in Results Section.

3 Results

We made a bioinformatics tool, focusing on a specific biological process. The tool can be used to query for orthologous proteins involved in cell size

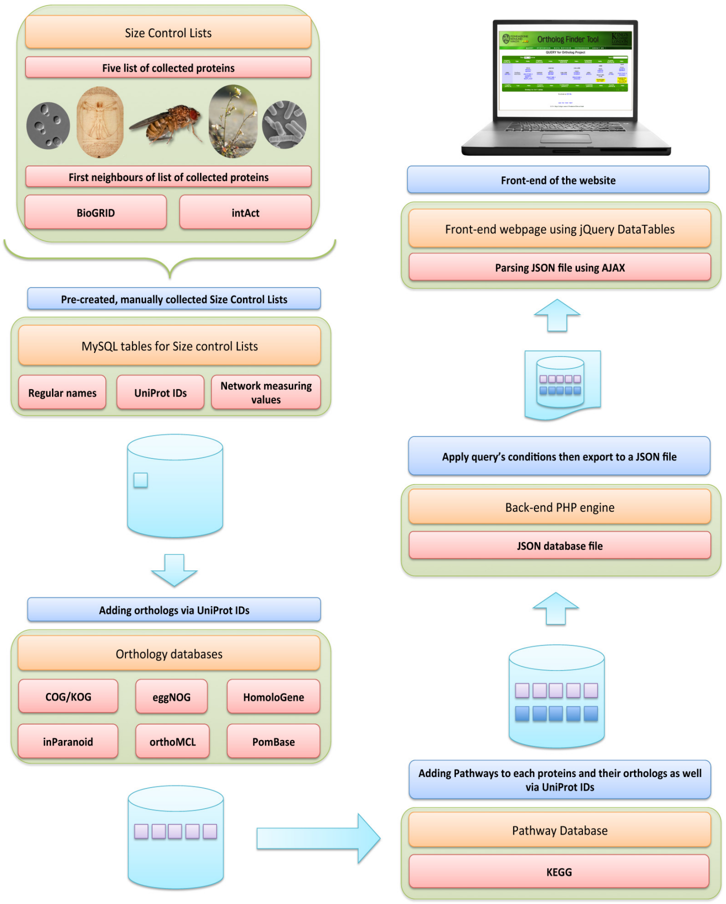
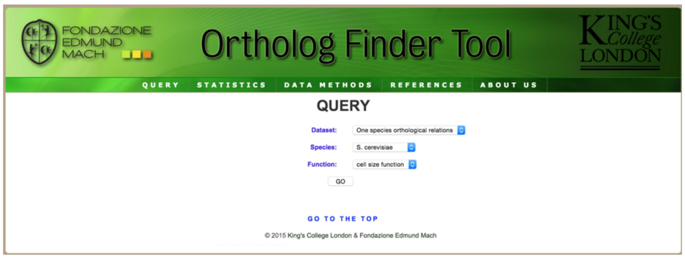

Size Control Lists
Five list of collected proteins
First neighbours of list of collected proteins
BioGRID
intAct
Pre-created, manually collected Size Control Lists
MySQL tables for Size control Lists
Regular names
UniProt IDs
Network measuring values
Adding orthologs via UniProt IDs
Orthology databases
COG/KOG
eggNOG
HomoloGene
inParanoid
orthoMCL
PomBase
Ortholog Finder Tool
Front-end of the website
Front-end webpage using jQuery DataTables
Parsing JSON file using AJAX
Apply query's conditions then export to a JSON file
Back-end PHP engine
JSON database file
Adding Pathways to each proteins and their orthologs as well via UniProt IDs
Pathway Database
KEGG

Figure 1: **Structure of the presented bioinformatics tool.** First the database selects the organism file with information on the regular name, UniProt ID, and network measuring values. Second the database matches the relevant orthologous proteins from 6 different sources via UniProt IDs. In addition each protein has biological pathway information. Subsequently a JSON file of the current query is created by the PHP engine. Finally the database front-end can be seen on the website in a single table format using jQuery plugins.

regulation. This tool handles functional proteins along with their PPI network's first neighbours. We integrated in this tool five annotated and fully sequenced organisms with extensive data on cell size regulation (*S. cerevisiae, S. pombe, H. sapiens, A. thaliana* and *D. melanogaster*) (see figure 3 for our tool structure).

The tool is able to display all orthologs of a given source gene from a given organism. One can also visualize whether this gene is a part of the specific cell size functional list in any of the other investigated organisms. Such a query can identify the proteins, which are appearing in all organisms as cell size regulators; therefore we can find the most conserved protein groups from these evolutionary distant organisms.

Currently there are 479 documented cell size regulator genes in budding yeast, 294 in fission yeast, 76 in human, 328 in Drosophila and 5 in Arabidopsis with an ortholog in all the other four species from which at least one is also a cell size control protein or first neighbour of such a protein in the PPI network.



QUERY

Dataset: One species orthological relations

Species: S. cerevisiae

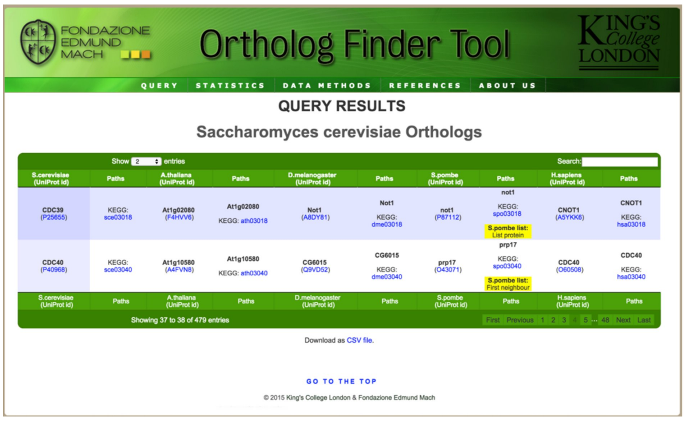
Function: cell size function

© 2015 King's College London & Fondazione Edmund Mach

Figure 2: **Interface of the database's Query page.** Users can select the query type and the organism which is used as a starting point in this page.

One can query for conserved group of proteins to identify the overlapping proteins among these 5 species. We found that there are 7 proteins that have a cell size regulator ortholog in all investigated organisms (see figure 4). These are Arabidopsis: HSL1, BAM2, T5P19_20; Drosophila: Akt1, Pkc53E; Budding yeast: SCH9, PKC1 (Human orthologs of LRRTM2 and PRKCB). These 7 proteins were found from 3 different starting points as there is no one-to-one orthology match between two organisms, rather the result depends on which gene in which specific organism was the starting point of our query. These proteins are sharing the common function of regulating morphogenesis and triggering cellular growth and remodelling pathways.

These seven genes from six orthologous sources identify the most conserved core genes in the regulation of cell size. However after careful observation one can see that starting from seven different proteins we always end up with the two common core genes. These are related to Protein Kinase C and the TOR pathway AGC family kinase, highlighting the existence of a conserved core of cell size regulation related to ribosome biogenesis through the TOR pathway [34, 35]. Target of Rapamycin (TOR) proteins are serine/threonine kinases that are controlled by environmental conditions and phosphorylate



QUERY RESULTS

Saccharomyces cerevisiae Orthologs

S.cerevisiae (UniProt id)	Paths	A.thaliana (UniProt id)	Paths	D.melanogaster (UniProt id)	Paths	S.pombe (UniProt id)	Paths	H.sapiens (UniProt id)	Paths
CDC39 (P25655)	KEGG: sce03018	At1g02080 (F4HVV6)	At1g02080 KEGG: ath03018	Not1 (A8DY81)	Not1 KEGG: dme03018	not1 (P87112)	not1 KEGG: spo03018 S.pombe list: List protein	CNOT1 (A5YKK6)	CNOT1 KEGG: hsa03018
CDC40 (P40968)	KEGG: sce03040	At1g10580 (A4FVN8)	At1g10580 KEGG: ath03040	CG6015 (Q9VD52)	CG6015 KEGG: dme03040	prp17 (O43071)	prp17 KEGG: spo03040 S.pombe list: First neighbour	CDC40 (O60508)	CDC40 KEGG: hsa03040
S.cerevisiae (UniProt id)	Paths	A.thaliana (UniProt id)	Paths	D.melanogaster (UniProt id)	Paths	S.pombe (UniProt id)	Paths	H.sapiens (UniProt id)	Paths

Showing 37 to 38 of 479 entries

Download as CSV file.

© 2015 King's College London & Fondazione Edmund Mach

Figure 3: **Interface of the database's Query Results page.** This is an example query for *S. cerevisiae* orthologs in other four organisms (*A. thaliana, D. melanogaster, S. pombe and H. sapiens*). There is a UniProt identifier next to each protein name if available. There are separate columns for biological pathway annotations from KEGG. Yellow textboxes indicate the orthologous protein appearance in other size control lists. UniProt IDs are linked to the UniProt database to find details on each gene.

several proteins. Studies in yeasts have identified that these proteins form complexes (TORC) and fill crucial position in the connection between metabolism and cell size regulation [36, 37].

Mammalian target of rapamycin complex 1 (mTORC1) affect the rate of transcription and ribosome biogenesis, through the regulation of RNA polymerases [38, 39, 40]. Ribosome biogenesis pathway is crucial in the maintenance of the rate of ribosome production and in general to control cell growth [40]. The ribosome content of a cell is important to a cell and has affect to the upper limit on the rate of protein synthesis. The mTORC1 complex of protein kinases is regulated by nutrients, anabolic hormones and oncogenic signalling pathways [39].

With our method we can point out evolutionally conserved pathways, which are crucial in cell size regulation. Furthermore we can determine some 'empty holes' where the cell size regulation function is conserved among 3 or 4 species but it has not been yet identified in the other 1 or 2 species. These holes could be investigated and the existing orthologs experimentally tested for their role in cell size regulation.

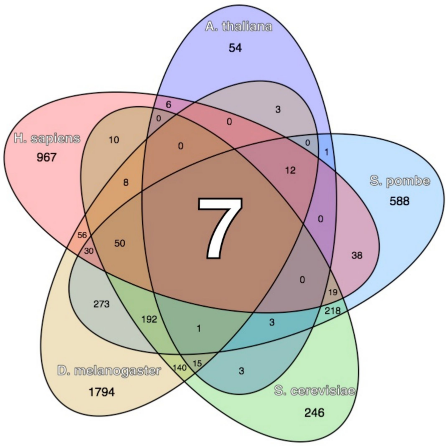


Figure 4: **Overlap in the cell size associated orthologs of five organisms.** The Venn-diagram illustrates the overlapping orthologs of conserved size-control proteins. The indicated numbers show how many orthologs of a specific gene from an intitial organism has been found in other cell size databases. The 7 proteins in the centre depict 2 proteins initially found from budding yeast, 2 proteins from fission yeast and 3 proteins from Arabidopsis

4 Discussion

In our integral approach we used 6 different orthological sources to find the conserved functional orthologous proteins of a given cell size regulator protein. These sources are differently collected functional orthological databases [1], nevertheless all of them aims the same goal to determine the closest functional orthologs of proteins. Users can find through a query all orthologs of an investigated protein together with the possible involvement of the orthologs in cell size regulation (see figure 2, yellow text boxes).

Our initial analysis found that the major key regulator pathway of cell size is the Ribosome Biogenesis pathway, which finding is in line with the current literature [13]. Moreover our analysis showed that there is a TOR pathway kinase, protein kinase C and their orthologs are cell size regulators in all five investigated organisms (including human).

We assume that with our comprehensive bioinformatics tool one can query appropriate functional orthologs among these organisms. Moreover users can display the biological functions investigated genes are associated with and visualize whick biological pathways (from KEGG) are associated with these genes. These allow users to identify key pathways and biological functions that might play crucial and conserved role in cell size regulation.

Our future plan is to add other biological functions and expand the number of organisms in our database. Our overarching aim is to create a unique tool to query functional conservancy across the phylogenetic tree in numerous biological functions. Current plan is to integrate some of the key Gene Ontology terms as a new function and check, which are the functionally conserved proteins that overlap among the investigated organisms.

References

[1] Fang G, Bhardwaj N, Robilotto R, Gerstein MB. Getting started in gene orthology and functional analysis. PLoS Comput Biol 2010;6. doi:10.1371/journal.pcbi.1000703.

[2] Fitch WM. Homology - a personal view on some of the problems. Trends Genet 2000;16:227-31. doi:10.1016/S0168-9525(00)02005-9.

[3] Cotton JA. Analytical methods for detecting paralogy in molecular datasets. Methods Enzymol 2005;395:700-24. doi:10.1016/S0076-6879(05)95036-2.

[4] O'Brien KP, Remm M, Sonnhammer ELL. Inparanoid: a comprehensive database of eukaryotic orthologs. Nucleic Acids Res 2005;33:D476-80. doi:10.1093/nar/gki107.

[5] Jensen LJ, Julien P, Kuhn M, von Mering C, Muller J, Doerks T, et al. eggNOG: automated construction and annotation of orthologous groups of genes. Nucleic Acids Res 2008;36:D250-4. doi:10.1093/nar/gkm796.

[6] Li L, Stoeckert CJ, Roos DS. OrthoMCL: identification of ortholog groups for eukaryotic genomes. Genome Res 2003;13:2178-89. doi:10.1101/gr.1224503.

[7] Li H, Coghlan A, Ruan J, Coin LJ, Hériché J-K, Osmotherly L, et al. TreeFam: a curated database of phylogenetic trees of animal gene families. Nucleic Acids Res 2006;34:D572-80. doi:10.1093/nar/gkj118.

[8] Thomas PD, Campbell MJ, Kejariwal A, Mi H, Karlak B, Daverman R, et al. PANTHER: A library of protein families and subfamilies indexed by function. Genome Res 2003;13:2129-41. doi:10.1101/gr.772403.

[9] Neufeld TP, de la Cruz AF, Johnston LA, Edgar BA. Coordination of growth and cell division in the Drosophila wing. Cell 1998;93:1183-93. doi:10.1016/S0092-8674(00)81462-2.

[10] Marshall WF, Young KD, Swaffer M, Wood E, Nurse P, Kimura A, et al. What determines cell size? BMC Biol 2012;10:1-22. doi:10.1186/1741-7007-10-101.

[11] Guertin DA, Sabatini DM. Cell size control. Encycl. Life Sci., Chichester, UK: John Wiley & Sons, Ltd; 2006. doi:10.1038/npg.els.0003359.

[12] Edgar BA. How flies get their size: genetics meets physiology. Nat Rev Genet 2006;7:907-16. doi:10.1038/nrg1989.

[13] Cook M, Tyers M. Size control goes global. Curr Opin Biotechnol 2007;18:341-50. doi:10.1016/j.copbio.2007.07.006.

[14] Jorgensen P, Tyers M. How cells coordinate growth and division. Curr Biol 2004;14:R1014-27. doi:10.1016/j.cub.2004.11.027.

[15] Ferrezuelo F, Colomina N, Palmisano A, Garí E, Gallego C, Csikász-Nagy A, et al. The critical size is set at a single-cell level by growth rate to attain homeostasis and adaptation. Nat Commun 2012;3:1-11. doi:10.1038/ncomms2015.

[16] Jorgensen P, Rupes I, Sharom JR, Schneper L, Broach JR, Tyers M. A dynamic transcriptional network communicates growth potential to ribosome synthesis and critical cell size. Genes Dev 2004;18:2491-505. doi:10.1101/gad.1228804.

[17] Yamamoto K, Gandin V, Sasaki M, McCracken S, Li W, Silvester JL, et al. Largen: a molecular regulator of mammalian cell size control. Mol Cell 2014;53:904-15. doi:10.1016/j.molcel.2014.02.028.

[18] Jorgensen P, Nishikawa JL, Breitkreutz B-J, Tyers M. Systematic identification of pathways that couple cell growth and division in yeast. Science 2002;297:395-400. doi:10.1126/science.1070850.

[19] Moretto F, Sagot I, Daignan-Fornier B, Pinson B. A pharmaco-epistasis strategy reveals a new cell size controlling pathway in yeast. Mol Syst Biol 2013;9:707. doi:10.1038/msb.2013.60.

[20] Hayles J, Wood V, Jeffery L, Hoe K, Kim D, Park H, et al. A genome-wide resource of cell cycle and cell shape genes of fission yeast. Open Biol 2013;3:130053. doi:http://dx.doi.org/10.1098/rsob.130053.

[21] Björklund M, Taipale M, Varjosalo M, Saharinen J, Lahdenperä J, Taipale J. Identification of pathways regulating cell size and cell-cycle progression by RNAi. Nature 2006;439:1009-13. doi:10.1038/nature04469.

[22] Hutchins JRA, Toyoda Y, Hegemann B, Poser I, Hériché J-K, Sykora MM, et al. Systematic analysis of human protein complexes identifies chromosome segregation proteins. Science 2010;328:593-9. doi:10.1126/science.1181348.

[23] Neumann B, Walter T, Hériché J-K, Bulkescher J, Erfle H, Conrad C, et al. Phenotypic profiling of the human genome by time-lapse microscopy reveals cell division genes. Nature 2010;464:721-7. doi:10.1038/nature08869.

[24] Graml V, Studera X, Lawson JLD, Chessel A, Geymonat M, Bortfeld-Miller M, et al. A Genomic Multiprocess Survey of Machineries that Control and Link Cell Shape, Microtubule Organization, and Cell-Cycle Progression. Dev Cell 2014;31:227-39. doi:10.1016/j.devcel.2014.09.005.

[25] Consortium TU. Activities at the Universal Protein Resource (UniProt). Nucleic Acids Res 2014;42:D191-8. doi:10.1093/nar/gkt1140.

[26] Cunningham F, Amode MR, Barrell D, Beal K, Billis K, Brent S, et al. Ensembl 2015. Nucleic Acids Res 2014:1-8. doi:10.1093/nar/gku1010.

[27] Stark C, Breitkreutz B-J, Reguly T, Boucher L, Breitkreutz A, Tyers M. BioGRID: a general repository for interaction datasets. Nucleic Acids Res 2006;34:D535-9. doi:10.1093/nar/gkj109.

[28] Orchard S, Ammari M, Aranda B, Breuza L, Briganti L, Broackes-Carter F, et al. The MIntAct project–IntAct as a common curation platform for 11 molecular interaction databases. Nucleic Acids Res 2014;42:D358-63. doi:10.1093/nar/gkt1115.

[29] Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, et al. Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome Res 2003;13:2498-504. doi:10.1101/gr.1239303.

[30] Coordinators NR. Database resources of the National Center for Biotechnology Information. Nucleic Acids Res 2014;42:D7-17. doi:10.1093/nar/gkt1146.

[31] Tatusov RL, Koonin E V, Lipman DJ. A genomic perspective on protein families. Science 1997;278:631-7. doi:10.1126/science.278.5338.631.

[32] Wood V, Harris M a, McDowall MD, Rutherford K, Vaughan BW, Staines DM, et al. PomBase: a comprehensive online resource for fission yeast. Nucleic Acids Res 2012;40:D695-9. doi:10.1093/nar/gkr853.

[33] Ogata H, Goto S, Sato K, Fujibuchi W, Bono H, Kanehisa M. KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res 1999;27:29-34. doi:10.1093/nar/27.1.29.

[34] Watanabe M, Chen CY, Levin DE. Saccharomyces cerevisiae PKC1 encodes a protein kinase C (PKC) homolog with a substrate specificity similar to that of mammalian PKC. J Biol Chem 1994;269:16829-36.

[35] Urban J, Soulard A, Huber A, Lippman S, Mukhopadhyay D, Deloche O, et al. Sch9 is a major target of TORC1 in Saccharomyces cerevisiae. Mol Cell 2007;26:663-74. doi:10.1016/j.molcel.2007.04.020.

[36] Wullschleger S, Loewith R, Hall MN. TOR signaling in growth and metabolism. Cell 2006;124:471-84. doi:10.1016/j.cell.2006.01.016.

[37] Martin DE, Soulard A, Hall MN, Rp PI. TOR Regulates Ribosomal Protein Gene Expression via PKA and the Forkhead Transcription Factor FHL1 2004;119:969-79.

[38] Mayer C, Grummt I. Ribosome biogenesis and cell growth: mTOR coordinates transcription by all three classes of nuclear RNA polymerases. Oncogene 2006;25:6384-91. doi:10.1038/sj.onc.1209883.

[39] Iadevaia V, Huo Y, Zhang Z, Foster LJ, Proud CG. Roles of the mammalian target of rapamycin, mTOR, in controlling ribosome biogenesis and protein synthesis 2012;40:168-72. doi:10.1042/BST20110682.

[40] Iadevaia V, Liu R, Proud CG. mTORC1 signaling controls multiple steps in ribosome biogenesis. Semin Cell Dev Biol 2014;36:1-8. doi:10.1016/j.semcdb.2014.08.004.

Tutorial - Design of artificial gene regulatory networks with fuzzy logic approach

Morgan Madec*[1], Abir Rezgui[1], Yves Gendrault[1,2], Christophe Lallement[1], Jacques Haiech[3]

[1] Laboratoire des Sciences de l'Ingénieur, de l'Informatique et de l'Imagerie (ICube), UMR 7357, Equipe SMH, 300 Boulevard Sébastien Brandt,
F-67412 Illkirch Cedex 02.

[2] ECAM Strasbourg-Europe - 2 rue de Madrid, F-67300 Schiltigheim.

[3] Laboratoire d'Innovation Thérapeutique (LIT), UMR 7200, 74 route du Rhin,
F-67400 Illkirch.

Abstract

In synthetic biology, automated synthesis of artificial gene regulatory networks achieving a biological fonction defined *a priori* is a hot research topic. Several approaches have already been demonstrated, mainly based on Boolean properties of such systems. However, Boolean descriptions are sometimes too rough and it becomes necessary to compile further information about the biological material involved in the construction ("strength" of the promoters, threshold of regulating proteins ...). In this case, the designer faces a trickier problem that has not (or only partially) been solved in engineering sciences, namely the adjustment of a large set of parameters in a predictive model with non-digital behavior with respect to a specified system's response. In this context, fuzzy logic is an intermediate level of description that tackles this design issue with a very interesting tradeoff between computation time and accuracy.

1 Gene regulatory network design automation

Synthetic biology, which aims at creating new biological functions by assembling artificial or natural biological parts, has been fast-growing over the past fifteen years [1]. This emerging science consists in several branches. In this tutorial, focus is put on the design of new artificial genetic regulatory networks. As it has been the case for microelectronics, the development of new technologies must be accompanied by the development of computer-aided design (CAD) tools that help the engineer during the whole design process. Since the beginning of the 2000s, several CAD tools for synthetic biology have been demonstrated. Most of them result from collaborative works between computer scientists and biologists [2-5]. One way to develop such tools consists in adapting existing tools, which have proven themselves their

*corresponding author: morgan.madec@unistra.fr

efficiency in microelectronics domain, for the biological context [5]. In digital microelectronics, several tool suites cover the whole design process, starting from the high-level specification and leading to the transistor-level schematic [6].

Unfortunately, in biology, the link between the abstracted Boolean description and the actual behavior of a gene regulatory network is not straightforward. Multi-valued logic approaches have already been investigated to bridge this gap [7-9]. In this tutorial, focus is put on an approach based on fuzzy logic [10]. The interest of this approach for the modeling of biological mechanisms as well as for system design automation has recently been demonstrated [11].

The purpose of this tutorial is to familiarize with this approach on several exercises. After a short introduction on the main concepts of fuzzy logic, the first exercise is a simple hand calculation that aims at understanding the way a fuzzy model is described and computed. Then, *ad hoc* Python scripts are used to perform more complex computations and to demonstrate that it is possible to describe a biological gate (AND gate described by Anderson *et al.* [12]) quite effectively with this approach. Finally, fuzzy logic is used for a gene regulatory network design purpose. The example used for this part is a band detector developed by Basu *et al.* [13].

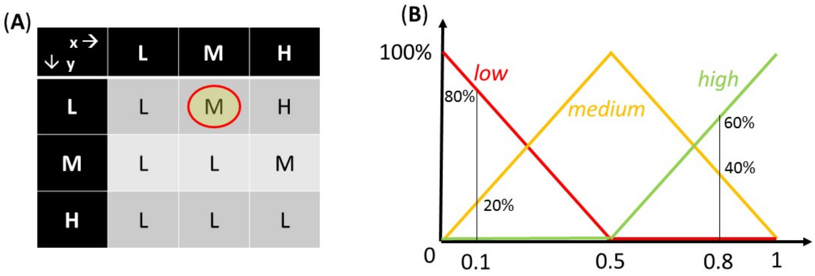
2 Overview of fuzzy logic

Boolean logic consists in describing a system with only two possible states: TRUE or FLASE. Fuzzy logic, introduced by Zadeh in 1965 [10], is an example of multi-valued logic (system can be described by more than two states) for which the continuous space is divided into intervals. By opposition to standard multi-valued logic methods, in fuzzy logic, a given input value does not correspond to a given interval but has a given degree of belonging (DoB) to all the intervals. In the following, those intervals are called membership functions (MFs). Most of the time, a linguistic variable is given to each MF (*e.g.* very low, medium, high, very high). Basically, the algorithm that computes fuzzy models can be divided into three main stages called fuzzyfication, rules evaluation and defuzzification. Fuzzification is a continuous-to-discrete domain conversion. The input data is converted into a vector which elements are the DoB of this data to all the MFs. Rules evaluation occurs in the discrete domain. It consists in computing the DoB of the output to each MFs as the function of input vectors according to logical rules (*e.g.* if A is medium and B is very-low then the output is very high). Finally, the defuzzification (or discrete-to-continuous domain conversion) consists in computing a continuous domain value from an output function obtained itself from the shape of output MFs and DoB of the output to these MFs. Details on these stages are illustrated on an example in the next part.

3 A simple hand calculation

Exercise: *Let us consider a system with two inputs (x and y) and one output (z). Each of them are normalized between 0 and 1 and the [0 ; 1] interval is divided in three regular triangle MFs (Fig. 1), namely low, medium and high. The rules of this system are also given as array in Fig. 1. Calculate the value of z when x = 0.8 and y = 0.1.*

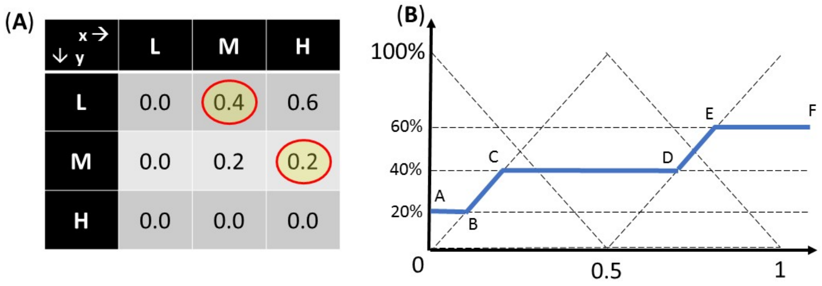
Fuzzification– The continuous domain inputs are discretized and converted into vectors containing the DoB to each MF. To simplify computations, inputs and outputs of the system are always normalized between 0 and 1. MFs are usually implemented as triangle function but other alternatives exist [10]. In our case, as the concentration of chemical species may vary over several decades, a logarithmic normalization is used [11]. Fuzzyfication consists in computing the DoB of each input to each MF. These values are recorded in an input vector. For example, if the normalized input is x = 0.8, according to Fig. 1, x belongs at 40% to MF "medium" and at 60% to MF "high". Thus, the input vector **X** = (0 ; 0.4 ; 0.6). By the same way, if y = 0.1, y belongs at 80% to the MF "low" and at 20% to the MF "medium". As a consequence, **Y** = (0.8 ; 0.2 ; 0).



x→ ↓ y	L	M	H
L	L	M	H
M	L	L	M
H	L	L	L

Figure 1: (A) Matrix of rules for the example described in section 3 (operon with one activator x and one repressor y). (B) Representation of the three membership functions.

Rules evaluation– Rules evaluation consists in evaluating the output vector as a function of the input vector according to logical properties or rules. They can be expressed either as a set of literal proposal or as a table called matrix of rules. In this case, each element of the matrix indicates the state of the output as a function of the state of the inputs (*e.g.* for the matrix given in Fig. 1, the highlighted element states that if x is medium and y is low, the output will be low). The representation of the rules in a matrix is an exhaustive description giving the output for every combination of inputs. It is a kind of truth table or a Karnaugh map used for Boolean algebra. In this case, a rule is a set of statements linked with the "and" word. Rules evaluation can be divided into two steps. First, the degree of realization (DoR) of each rule is computed according to the input vector. Most of the time, the DoR for each rule is defined as the minimum DoB of each statements of the rule. Let us consider the highlighted element in the matrix of rules (Fig. 1). The DoB of x to "medium" is 40 % and the DoB of y to "low" is 80 %. Thus, the DoR of this rules is 40 %. The results on every rules of the example are given on the Fig. 2. Now, the second step consists in computing the output vector (DoB of the output to each output's MF). In the matrix of rules, several rules may lead to the same output state. For a given output state, the DoB corresponds to the maximum of the DoR over all the rules that lead to the given MF. In our example, a "medium" output can be obtained both with a "medium" x and a "low" y or with a "high" x and a "medium" y. The DoRs of this rules are respectively 40 % and 20 %. The DoB of the output to "medium" is thus 40 %. By this way, the output vector can be computed: Z = (0.2 ; 0.4 ; 0.6). It should be noticed that for this two steps, the min/max may be replaced by more sophisticated function [10].



(A)

x → ↓ y	L	M	H
L	0.0	0.4	0.6
M	0.0	0.2	0.2
H	0.0	0.0	0.0

Figure 2: (A) Degree of realization of each rule of the example. (B) Output function of the modeled system.

Defuzzification– Deffuzification consists in converting the output vector **Z** into a value z which corresponds to the centroid of the output function $f(a)$. The output function is defined as following:

$$f(a) = \max_k[\min(MF_k(a), Z(k))]$$

where $MF_k(a)$is the k-th output MF and Z(k) is the k-th element of the output vector. The centroid is computed as following:

$$z = \frac{\int_0^1 a \cdot f(a) \cdot da}{\int_0^1 f(a) \cdot da}$$

The bisector method is an approximation of the centroid method [10]. Bisector consists in computing to the z value which separates the area below the output function into two equal parts. Its main asset over centroid method is that it can be computed with geometrical considerations and avoid numerical integrations. In our example, the output function is given in Fig. 2. Coordinates of the points A to F are respectively (0 ; 0.2), (0.1 ; 0.2), (0.2 ; 0.4), (0.7 ; 0.4), (0.8 ; 0.6) and (1 ; 0.6). Thus, the area under the AB, BC, CD, DE and EF segments are respectively 0.02, 0.03, 0.20, 0.05 and 0.12. The total area being 0.42, the bisector is on the CD segment and the area below the CD segment and on the left of the bisector must be 0.16. As a consequence, one can point out that z = 0.6.

Under the condition and with the methods exposed in this part (fixed triangle MFs, min/max rule evaluation, bisector defuzzification), the computation of a fuzzy model can be integrally performed with simple geometrical consideration and its implementation on a computer becomes faster. We recently demonstrated a C-implementation of the fuzzy algorithm for which computation time is about 500 ns per point for 5 MFs and has a linear increase with the number of MFs [11].

4 Fuzzy logic for gene regulatory network modeling

Fuzzy logic can be used for modeling purpose. In this case, inputs sweep the [0 ; 1] interval and the output is computed for each input combination. The Python script Anderson.py computes the response of any fuzzy model of a 2-input 1-output gene regulatory network modeled by a 5x5 matrix of rules (5 MF per input/output) and compares the simulation result to actual data (in this case, the response of a biological AND gate designed by Anderson *et al.* [12]). The likelihood criterion is the mean square error.

Exercise: *Let us consider the biological AND gate designed by Anderson* et al. *(Fig. 3). Use the Anderson.py graphical user interface in order to find the matrix of rules that best fits normalized Anderson's gate response. The answer of given on Fig. 4.*

We developed an algorithm that automatically performs this investigation and measures the mean square error between simulation results of the optimized model and actual data for several numbers of MFs. Results are given in Fig. 5. Two main conclusions can be drawn from this analysis: i) Fuzzy logic remains inaccurate even if the number of MFs increases and ii) 5 or 7 MFs seems to be a good tradeoff between model accuracy and simulation time.

5 Fuzzy logic for gene regulatory network automated design

Fuzzy logic can also be used for design automation. In this case, for each device of the system, an algorithm search which matrix of rules should be used in order to get as close as possible to the response of the system defined

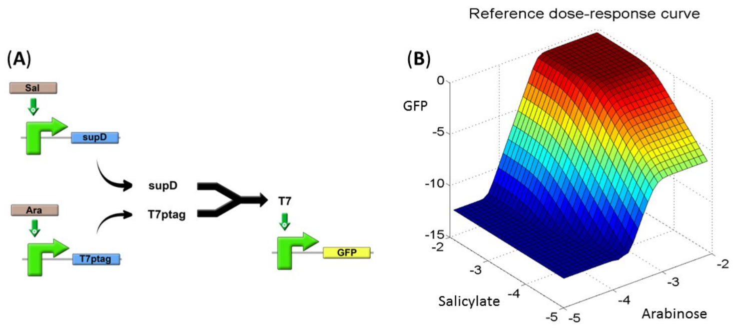
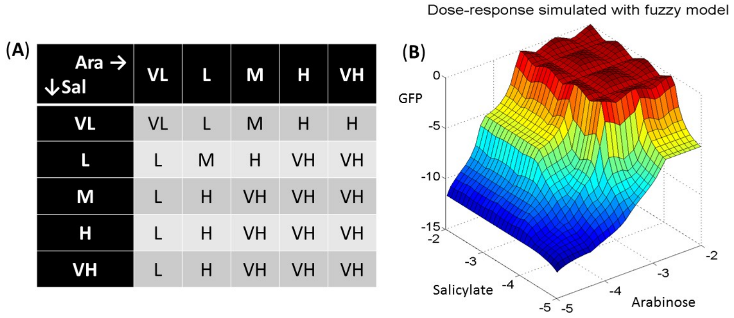


Figure 3: (A) Anderson's AND gate. Arabinose and salicylate activates respectively the synthesis of supD and T7ptag which bind each other to obtain T7, an activator for the promoter carrying the gene coding for the GFP. (B) Measured response of the system.



Ara → ↓Sal	VL	L	M	H	VH
VL	VL	L	M	H	H
L	L	M	H	VH	VH
M	L	H	VH	VH	VH
H	L	H	VH	VH	VH
VH	L	H	VH	VH	VH

Figure 4: (A) Matrix of rules that best fits the measured data obtain with automated research algorithm. (B) Simulation of the fuzzy model with the matrix of rules given in (A).

a priori. By this way, several useful clues can be obtained to guide the system design (strength of the promoters, threshold concentration of regulating proteins, slope of the dose-response curve ...). In practice, a library (FuzzyLib) which contains a set of possible matrices of rules is build. Each matrix of rules corresponds to one Boolean behavior. Only five of them are considered: i) a inducible promoter with an activator (Buffer); ii) a inducible promoter with two activators (OR gate); iii) a constitutive promoter with one repressor (NOT gate); iv) a constitutive promoter with two repressors (NOR gate) and v) a inducible promoter with one activator and one repressor (INH gate). Finally, in the FuzzyLib, there are 6 declinations of Buffer and NOT gate and 36 declination for the others 2-input gates. To illustrate the purpose, the 6 declinations of the NOT gate are given in Fig. 6.

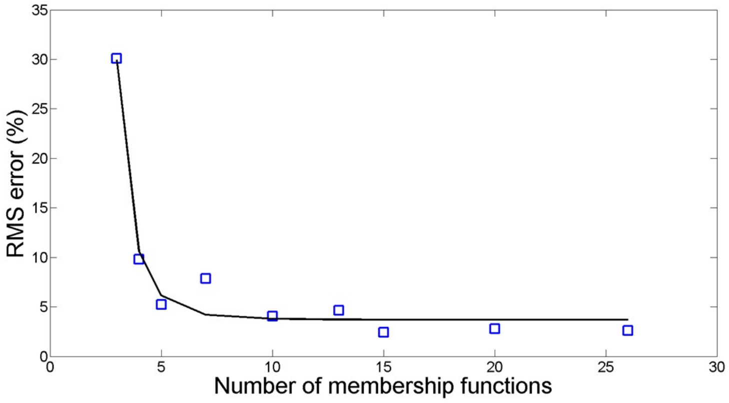


Figure 5: RMS error of between fuzzy model and actual data for Anderson's AND gate as a function of the number of membership functions used in the fuzzy model.

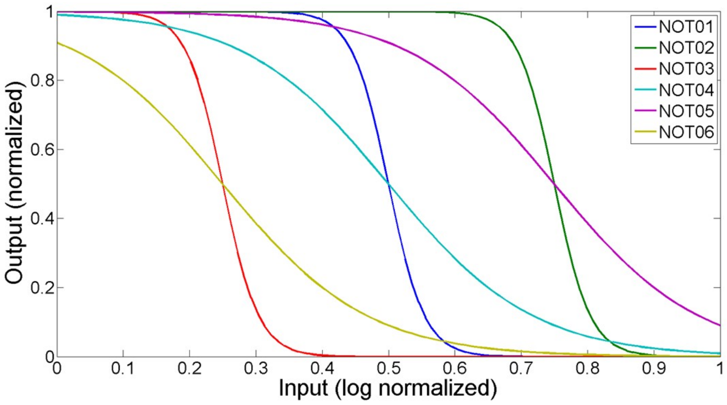


Figure 6: Six declinations of the biological NOT gate (constitutive promoter with a repressor). Each curve corresponds to a given threshold concentration and slope.

In the following, the design automation process is illustrated on a band detector designed by Basu *et al.* [13]. This gene regulatory network synthesize GFP as soon as the concentration of AHL (acyl homoserine lactone) is comprised in a given range. The construct is depicted in Fig. 7. It is composed with 4 genes (2 buffers sharing the same operon, 1 NOT gate and 1 NOR gate). The Python script Basu.py gives the possibility to browse into the FuzzyLib for the three involved devices and plots the dose-response corresponding to each device as well as the response of the complete system.

Exercise: *Use the Basu.py graphical user interface in order to browse the FuzzyLib library and find which matrices of rules are the most suitable to* *get as close as possible to the targeted system response. Deduce from this investigation properties of the promoter that should be used in the actual gene regulatory network.*

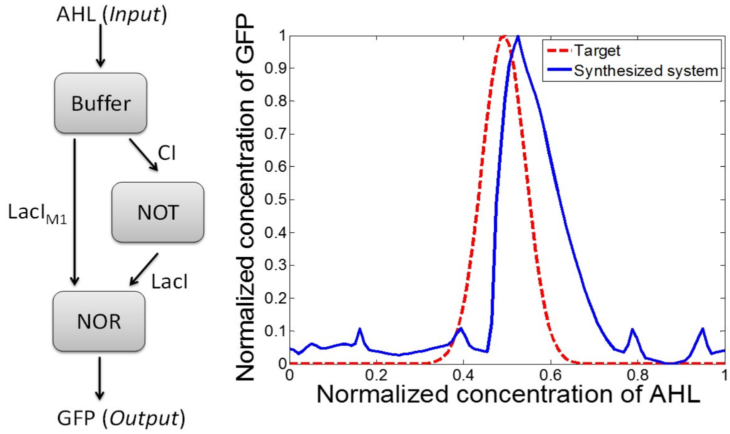


Figure 7: (A) Basu's band detector composed with three Boolean functions (a buffer with two outputs, an inverter and a NOR gate). (B) Targeted response and fuzzy simulation result with the best set of matrices of rules suggested by an exhaustive research algorithm.

For this example, there is only 1,300 possible combinations of matrices of rules. For more complex systems, an efficient algorithm that automatically finds the best combination is still under construction. Up to now, the only algorithm that has been implemented is an exhaustive research algorithm which tests all the combinations and returns the one that exhibit the least mean square error in comparison with the targeted response. In the case of Basu's band detector, the algorithm suggests using a inducible promoter with a high threshold for the Buffer (a large quantity of activator is required to activate the transcription), a constitutive promoter with a strong repressor (a low concentration of repressor is sufficient to repress the promoter) and an asymmetric NOR gate (both activators do not have the same strength).

Exhaustive research algorithm has strong limitations. The most important is the computation time: on a more complex example (XOR gate designed by Terzer *et al.* [14]), up to 3 million of combinations have to be tested and the calculation time exceeds 20 minutes. An improved algorithm, which consist in testing only the combinations that seem to be relevant *a priori*, reduces the computation time by a factor of 10. Some other improvements are still under investigation. They are based on standard optimization algorithms (simulated annealing, particle swarm, genetic programming...).

Once the best combination has been identified, our software returns a text file with the chosen matrices of rules as well as a SBML [15] description of the system that can be imported in Copasi [16] to perform further analysis. For this conversion, each device is modeled with standard dynamic models (based on differential equations) which parameters are adjusted to different default values depending on the selected matrices of rules. The Python script used for this tutorial also integrates this feature.

Exercise: *Use Basu.py to export the SBML model corresponding to your best combination of matrices of rules and simulated the system with Copasi.*

6 Conclusion

This tutorial was to introduce the concept of fuzzy logic applied to synthetic biology. All the tools presented are now available in two versions, one developed in Python for educational purpose and the other developed and optimized in C-language. Among the tools we discussed, the automated gene regulatory networks synthesizer has undoubtedly the most potential. It takes as input an assembly of Boolean biological gates (gate-level schematic) and the targeted system's response. However, it can also be coupled with an upstream digital synthesizer [5] that provides the gate-level schematic from a description at a higher level of abstraction. To make it fully operational, two aspects need further investigations. First, the algorithm for selecting the optimal set of matrices of rules must be improved to reduce the computation time. Second, the FuzzyLib must be completed and linked to actual biological material. By this way, we would have a list of the operons that can be used for each matrix of rules in order to construct directly the genetic regulatory network from the result of the optimization process. The tool would also move from a computer-aided design tool that guides the choices of the bio-designer into a true genetic design automation tool.

Acknowledgments

The authors would like to thank Vincent Wlotzko, Jean-Pascal Caussé, Martin Lemaire and Pamina Bernou, students of Telecom Physique Strasbourg, for their valuable contribution to this work and the preparation of this tutorial.

References

[1] D. Endy, *Foundations for engineering biology*, Nature, vol. 438, pp.449–453, 2005.

[2] J. Beal et al, *An End-to-End Workflow for Engineering of Biological Networks from High-Level Specifications*, ACS Synthetic Biology, vol. 1(8). 2012.

[3] D. Chandran *et al. TinkerCell: modular CAD tool for synthetic biology*, Journal of Biological Engineering, vol. 3(19), 2009. `http://www.tinkercell.com`

[4] M.J. Czar, *Writing DNA with GenoCAD*, Nucleic acids research, vol. 37, 2009. http://www.genocad.org

[5] M. Madec et al, *EDA inspired open-source framework for synthetic biology* Proc. IEEE BioCAS, Rotterdam (NL), 30 Oct – 2 Nov 2013

[6] E. Brunvand, *Digital VLSI Chip Design with Cadence and Synopsys CAD Tools*, ed. Addison Wesley, 2009.

[7] C. Chaouiya, *Qualitative modelling of biological regulatory networks combining a logical multi-valued formalism and Petri nets*, in Discrete event workshop, WODES 2008

[8] R. Franke, *From binary to munti-valued to continuous models: the lac operon as a case study*, Journal of integrative bioinformatics, vol. 7(1), 2010

[9] G. Bernot, *Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic*, Journal of theoretical biology, vol. 229(3), 2004.

[10] L.A. Zadeh, *Fuzzy Sets*, Information and Control, vol. 8, pp. 338–353, 1965

[11] Y. Gendrault *et al.*, *Using fuzzy logic in synthetic biosystems design*, Proc. IEEE BioCAS 2014, Lausanne (Switzerland), 22–24 Oct. 2014.

[12] J.C. Anderson *et al.*, *Environmental signal integration by a modular AND gate* Molecular Systems Biology, vol. 3, 2007

[13] S. Basu *et al.*, *A synthetic multicellular system for programmed pattern formation*, Nature, vol. 434, 2005

[14] Terzer, M. *et al.*, *Design of a biological half adder*, Synthetic Biology, IET , vol.1, no.1.2, pp.53–58, 2007.

[15] M. Hucka, *The system biology markup language (SBML): a medium for representation and exchange of biochemical network models*, Bioinformatics, vol. 19(4). 2003. `http://sbml.org`.

[16] S. Hoops, *COPASI – a COmplex PAthway SImulator*, Bioinformatics, vol. 22(24). 2006. `http://www.copasi.org`

The mitochondrion metabolic model iAS253 revisited.

Anna Zhukova[1] and Jean-Pierre Mazat[1]

[1] IBGC- CNRS UMR 5095, 1 rue Camille Saint Saens, CS 61390,
33077 Bordeaux Cedex, France

Abstract

Mitochondria are important organelles of a eukaryotic cell, a source of cellular energy, involved in the generation of ATP. Mitochondria are implicated in several human diseases, and understanding mitochondrial metabolism may help finding therapeutic actions.

Using the metabolic model iAS253 by Robinson and Smith, we apply the constraint-based analysis (FBA) techniques to study possible metabolic benefits in the case of fumarase deficiency.

1 Introduction

Mitochondria are involved in many essential metabolic processes, thus their defects can cause a wide range of human pathologies [1].

Metabolic modelling is a tool to understand the underlying metabolism changes and to propose possible therapeutic actions. Several models describing human metabolism were created, including large-scale ones [2, 3, 4] and those with a particular focus on the mitochondrion metabolism [5, 6, 7, 8, 9]. In this work we have chosen to use the model iAS253 [9] as it focuses on mitochondrion and is the most complete among the existing models of mitochondrial metabolism. As in [9], we approach fumarase deficiency and study metabolic ways to overcome this defect in different type of tissues characterized by different inputs in mitochondrial metabolism.

The model iAS253 describes mitochondrial metabolism in human's heart. It contains three compartments (extracellular, cytosol and mitochondrion), 253 reactions, 245 metabolites and 89 transport steps across the mitochondrial membrane. The identifiers of reactions and metabolites correspond to the entries in the KEGG reaction and the KEGG compound databases [10]. The model does not contain any global parameters; the local parameters defined in the model describe the reaction fluxes and flux constraints, needed for constraint-based analysis. A schematic representation of the model is shown in Figure 1.

1.1 Constraint-based analysis

Metabolic phenotypes can be defined in terms of flux distributions through a metabolic network. *Dynamic analysis* of metabolic flux distributions require

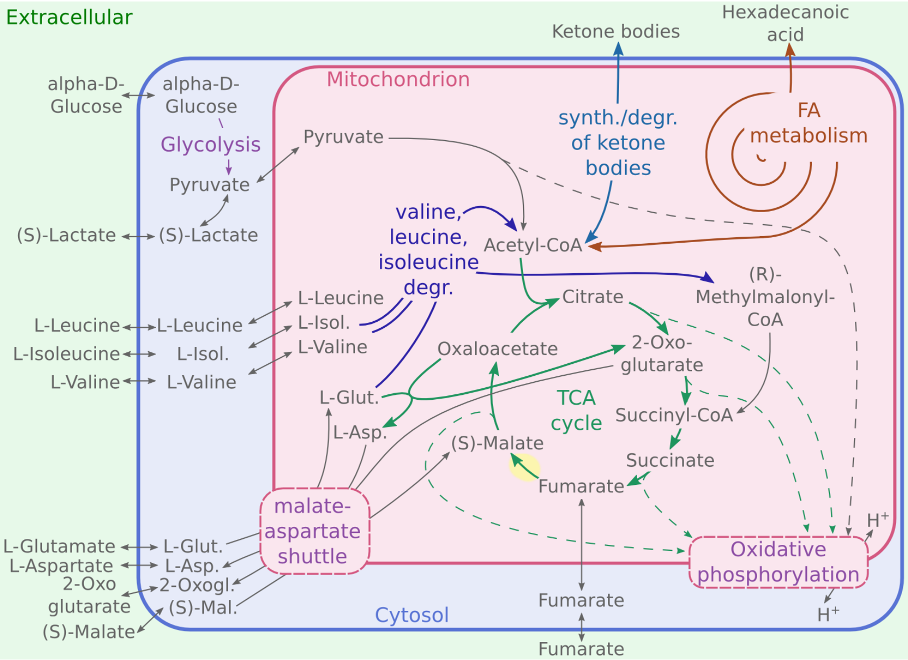


Figure 1: **Model iAS253.** Model iAS253 focusses on the processes operating in the mitochondrion of a human heart. The main pathways are shown in different colours: *TCA cycle* (green), *fatty acid metabolism* (orange), *valine, leucine and isoleucine degradation* (blue), *synthesis and degradation of ketone bodies* (violet).

kinetics and concentration information about enzymes and various cofactors. For metabolic networks that lack this information the *constraint-based* modelling procedure [11, 12] is well-adapted. It does not strive to find a single solution but rather finds a collection of all allowable solutions to the governing equations that can be defined (a solution space). The solutions that violate any of the imposed constraints are excluded from the solution space. The constraints include stoichiometry, reversibility of reactions, and enzymatic capacity [13].

The *flux balance analysis (FBA)* [14] is an example of a constrain-balance analysis. FBA defines an objective function relevant to the studied problem (e.g. ATP production in mitochondria) and finds a flux distribution that optimises (e.g. maximises) the objective function at steady state. FBA finds only one of the possible solutions. To circumvent this problem, one can use the *flux variability analysis (FVA)* [15, 16, 17], which estimates the maximum and minimum values of all the fluxes that will satisfy the constraints while reaching the same optimal value of the objective function.

There are several software tools that perform FBA and FVA, examples of which include the constraint-based reconstruction and analysis (COBRA) toolbox [18] for MATLAB; FAME, the web-based flux analysis and modeling environment [19]; and COBRApy [20], a COBRA toolbox for Python. In our study we use COBRApy.

2 Modifications to the model

To evaluate the model iAS253 we run FBA under different conditions. The model iAS253 defines 6 pseudo reactions intended to be used as objective functions for flux analysis: amino acids for protein synthesis, nucleotides for DNA and for RNA synthesis, lipids for lipid synthesis, and the production of ATP and of haem. In our study we use only the ATP production as the objective function.

2.1 Futile cycles

FBA under normal conditions finds a solution with an average flux absolute value of 29 $\mu mol/min/gDW$, while for five reactions shown in Figure 2 the flux value is much larger and reaches the extreme value allowed by their constraints (± 1000): *ATP:GDP phosphotransferase* (R00330MM); *ATP:AMP phosphotransferase* (R00127MM); *UTP:pyruvate 2-O-phosphotransferase* (R00659MM); *UTP:AMP phosphotransferase* (R00157MM); *GTP:pyruvate 2-O-phosphotransferase* (R00430MM)[1]. FVA shows that in the solution space corresponding to the maximal ATP production, the fluxes through these reactions are not constrained, i.e. can have any value between -1000 and 1000. It suggests a presence of a futile cycle. Moreover, *GTP:pyruvate 2-O-phosphotransferase* is irreversible with $\Delta G_0 = -30.1\ kJ$ [21]. We updated the bounds of this reaction, which made it irreversible and eliminated the futile cycle.

2.2 Pathway annotations

The reactions of the model iAS253 use the identifiers of the entries in the KEGG reaction database. KEGG provides a REST API that permits automatic extraction of information from KEGG databases. To annotate the model with pathways we extracted a list of human pathways from the KEGG pathway database and a list of reactions that correspond to each pathway from the KEGG reaction database. For each pathway we calculated the ratio of its reactions that are present in the model to the total number of reactions listed for this pathway in KEGG. For the ratios larger than 0.5 (more than a half of the

[1] The reaction names are taken from the KEGG reaction database [10].

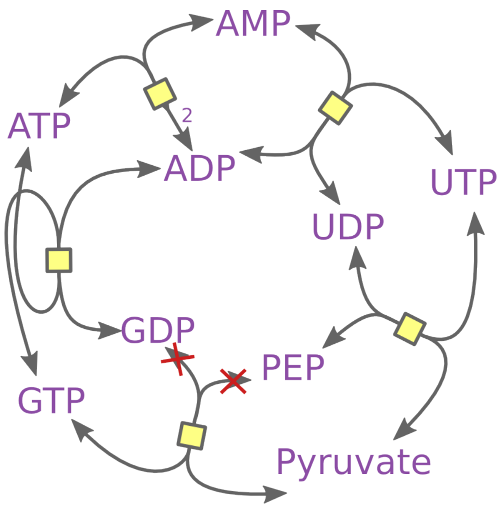


Figure 2: **Futile cycle of NTP ↔ NDP interconversion.** The cycle consists of five reactions (yellow squares) that are reversible in the original iAS253 model. We updated the bounds of the *GTP:pyruvate 2-O-phosphotransferase* reaction (red), which made it irreversible and eliminated the futile cycle.

pathway was found in the model), we assumed that the corresponding pathway is present in the model. This procedure allowed us to annotate the model with 4 pathways and 3 sub-pathways (see Figure 1):

1. *citrate cycle (TCA cycle)* (KEGG identifier: hsa00020);
2. *fatty acid metabolism* (hsa01212) and 3 sub-pathways:
   (a) *fatty acid biosynthesis* (hsa00061);
   (b) *fatty acid elongation* (hsa00062);
   (c) *fatty acid degradation* (hsa00071);
3. *valine, leucine and isoleucine degradation* (hsa00280);
4. *synthesis and degradation of ketone bodies* (hsa00072).

In our study we omit the more generic *fatty acid metabolism* pathways and instead consider the 3 more specific sub-pathways. The *fatty acid elongation* and *fatty acid degradation* pathways have a large intersection (18 (reversible) reactions out of 28 and 33 respectively) with the opposite directions of the fluxes through these reactions.

3 Simulation of fumarase deficiency

3.1 Normal conditions

Under normal conditions (inputs: 0.9 $\mu mol/min/gDW$ of *glucose*, 0.575 $\mu mol/min/gDW$ of *lactate* and 0.412 $\mu mol/min/gDW$ of *fatty acids*) to obtain the optimal ATP production (of 139.42 $\mu mol/min/gDW$) a flux of 6.97 $\mu mol/min/gDW$ through the fumarase reaction (R01082MM, highlighted yellow in Figure 1) is required. Non-null fluxes are found through the following pathways: *citrate cycle (TCA cycle)*; *valine, leucine and isoleucine degradation*; *synthesis and degradation of ketone bodies*; *fatty acid degradation*. The *fatty acid biosynthesis* and *fatty acid elongation* pathways are not active.

3.2 Varying fumarase flux constraints

To study the fumarase deficiency we first run a simulation similar to the one described by Smith and Robinson.

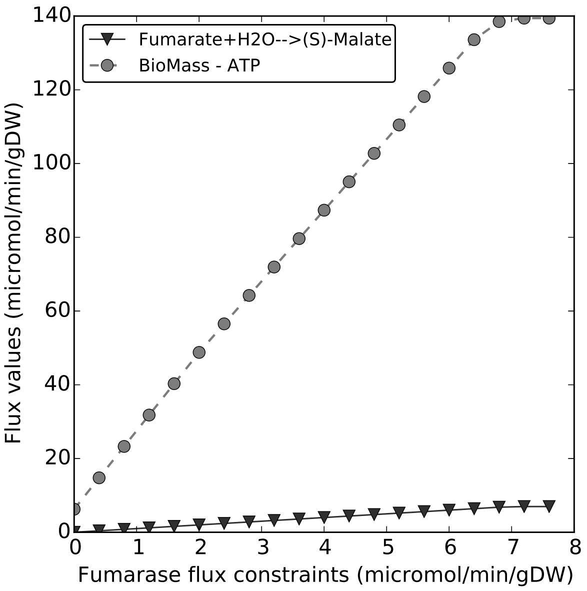
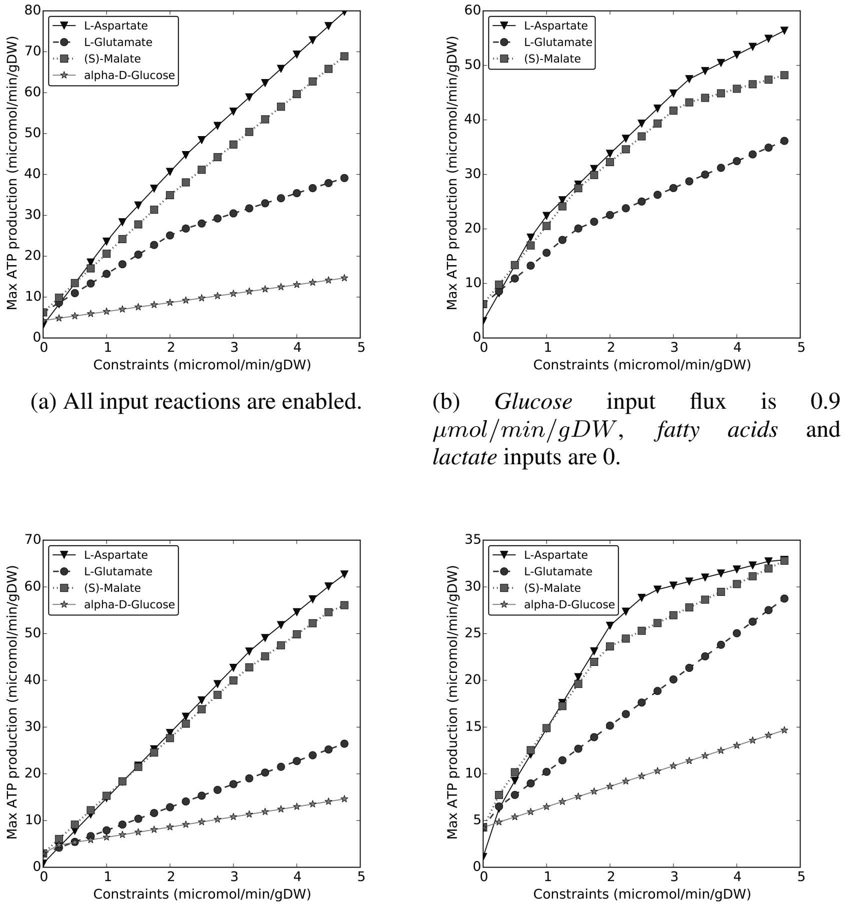


Figure 3: **Effect of varying fumarase flux on maximal ATP production.** 6.97 $\mu mol/min/gDW$ is the value of the fumarase flux necessary for the maximal ATP production (139.42 $\mu mol/min/gDW$), and after reaching this value the fumarase reaction stabilises.



(a) All input reactions are enabled.

(b) *Glucose* input flux is 0.9 $\mu mol/min/gDW$, *fatty acids* and *lactate* inputs are 0.

(c) *Fatty acids* input flux is 0.412 $\mu mol/min/gDW$, *glucose* and *lactate* inputs are 0.

(d) *Lactate* input flux is 0.575 $\mu mol/min/gDW$, *glucose* and *fatty acids* inputs are 0.

Figure 4: **Effect of input reactions on maximal ATP production when fumarase is impaired.** Effect of changing the constraints of input reactions on maximal ATP production (all the curves are independent) when the fumarase reaction is 0, under different conditions: (a) all input reactions are allowed (general case as in Figure 3) (b) the *fatty acids* and *lactate* input reactions are impaired, the input flux of the *glucose* reaction is fixed at 0.9 $\mu mol/min/gDW$; (c) the *glucose* and *lactate* input reactions are impaired, the flux of the *fatty acids* input reaction is fixed to 0.412 $\mu mol/min/gDW$; (d) the *fatty acids* and *glucose* input reactions are impaired, the flux of the *lactate* input reaction is fixed to 0.575 $\mu mol/min/gDW$.

We changed step by step the constraints for the flux through the fumarase reaction: $-i \leq v_{fum} \leq i$, from 0 (i.e. $i = 0$, fumarase knocked-out) to 8 $\mu mol/min/gDW$(to allow the optimal flux value) and run FBA to optimise the ATP production. The resulting curves are shown in Figure 3. Smith and Robinson describe a similar simulation (Figure 1 in [9]), but instead of changing the flux constraints, they impose the value of the flux. For the flux not greater than 6.97 $\mu mol/min/gDW$ both simulations give the same result, as for maximal ATP production, the maximal allowed flux though fumarase is necessary. However, for the imposed flux values that are greater than the optimal value 6.97, the rest of the system cannot produce enough fumarate/succinate without lowering the ATP production, thus the ATP Biomass curve in Smith's and Robinson's study (Figure 1 in [9]) goes down, while the one in Figure 3 stabilises together with the flux through fumarase.

3.3 Impaired fumarase reaction

To study the effect of each metabolite input from the environment on maximal ATP production under conditions when the fumarase reaction was knocked out, we run a simulation with all the inputs allowed and three independent simulations with each input at its flux value used in the general study (Figure 4) with the other inputs equal to zero. In all cases the *citrate cycle (TCA cycle)*, and *valine, leucine and isoleucine degradation* pathways were active.

When all the three inputs are allowed, the knock out of the fumarase reaction reduces the maximal ATP production from 139.42 to 6.26 $\mu mol/min/gDW$. Under these conditions *synthesis of ketone bodies* and *fatty acid elongation* pathways are active. However, the change of the boundary constraints for several metabolite import reactions (*aspartate*, *glutamate*, *malate* and *glucose*) can increase maximal ATP production flux value (see Figure 4 (a)).

When only *glucose* input was allowed, the maximal ATP production flux was 6.20 $\mu mol/min/gDW$ and there were a *synthesis of ketone bodies* and a *fatty acid elongation* (as when all input fluxes were present). When only the input of *lactate* was allowed the maximal ATP production flux dropped to 4.25 $\mu mol/min/gDW$, none of the *fatty acid metabolism* pathways were active, and instead of synthesis, the *degradation of ketone bodies* took place.

In the case when only the *fatty acids* input was active, the maximal ATP production flux went to 2.96 $\mu mol/min/gDW$, the *degradations of fatty acids* and of *ketone bodies* occurred.

In all cases an increase in *aspartate* and/or *glutamate* inputs through the *malate-aspartate shuttle* have a significant effect on ATP synthesis, contrary to an increase in glucose input which only leads to a very low augmentation in ATP synthesis.

4 Conclusion

We have used the slightly modified model iAS253 by Robinson and Smith to simulate fumarase deficiency under various conditions. In the future work it would be interesting to investigate other mitochondrial disorders such as (partial) ATP syntase deficiency.

The model could be further improved by refining the reaction boundary constraints, for example, based on reaction Gibbs energies [23].

Acknowledgements

The authors thank AFM and ITMO Cancer (AVIESAN) and INCa for financial support.

References

[1] Schapira AHV, Mitochondrial disease. *Lancet* 2006, **368**(9529):70–82.

[2] Zhao Y, Huang J, Reconstruction and analysis of human heart-specific metabolic network based on transcriptome and proteome data. *Biochemical and biophysical research communications* 2011, **415**(3):450–4.

[3] Karlstädt A, Fliegner D, Kararigas G, Ruderisch HS, Regitz-Zagrosek V, Holzhütter HG, CardioNet: a human metabolic network suited for the study of cardiomyocyte metabolism. *BMC systems biology* 2012, **6**:114.

[4] Thiele I, Swainston N, Fleming RMT, Hoppe A, Sahoo S, Aurich MK, Haraldsdottir H, Mo ML, Rolfsson O, Stobbe MD, Thorleifsson SG, Agren R, Bölling C, Bordel S, Chavali AK, Dobson P, Dunn WB, Endler L, Hala D, Hucka M, Hull D, Jameson D, Jamshidi N, Jonsson JJ, Juty N, Keating S, Nookaew I, Le Novère N, Malys N, Mazein A, Papin JA, Price ND, Selkov E, Sigurdsson MI, Simeonidis E, Sonnenschein N, Smallbone K, Sorokin A, van Beek JHGM, Weichart D, Goryanin I, Nielsen J, Westerhoff HV, Kell DB, Mendes P, Palsson BO, A community-driven global reconstruction of human metabolism. *Nature Biotechnology* 2013, **31**(5):419–25.

[5] Ramakrishna R, Edwards JS, McCulloch A, Palsson BO, Flux-balance analysis of mitochondrial energy metabolism: consequences of systemic stoichiometric constraints. *American journal of physiology. Regulatory, integrative and comparative physiology* 2001, **280**(3):R695–704.

[6] Vo TD, Greenberg HJ, Palsson BO, Reconstruction and functional characterization of the human mitochondrial metabolic network based on proteomic and biochemical data. *The Journal of biological chemistry* 2004, **279**(38):39532–40.

[7] Yugi K, Tomita M, A general computational model of mitochondrial metabolism in a whole organelle scale. *Bioinformatics (Oxford, England)* 2004, **20**(11):1795–6.

[8] Thiele I, Price ND, Vo TD, Palsson BO, Candidate metabolic network states in human mitochondria. Impact of diabetes, ischemia, and diet. *The Journal of biological chemistry* 2005, **280**(12):11683–95.

[9] Smith AC, Robinson AJ, A metabolic model of the mitochondrion and its use in modelling diseases of the tricarboxylic acid cycle. *BMC systems biology* 2011, **5**:102.

[10] Kanehisa M, Goto S, Sato Y, Furumichi M, Tanabe M, KEGG for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Res.* 2012, **40**(Database issue):D109–14.

[11] Bonarius HP, Schmid G, Tramper J, Flux analysis of underdetermined metabolic networks: the quest for the missing constraints. *Trends in Biotechnology* 1997, **15**(8):308–314.

[12] Edwards JS, Covert M, Palsson B, Metabolic modelling of microbes: the flux-balance approach. *Environmental microbiology* 2002, **4**(3):133–40.

[13] Reed JL, Palsson BO, Thirteen years of building constraint-based in silico models of Escherichia coli. *Journal of bacteriology* 2003, **185**(9):2692–9.

[14] Orth JD, Thiele I, Palsson BO, What is flux balance analysis? *Nature biotechnology* 2010, **28**(3):245–8.

[15] Burgard AP, Vaidyaraman S, Maranas CD, Minimal reaction sets for Escherichia coli metabolism under different growth requirements and uptake environments. *Biotechnology progress* 2001, **17**(5):791–7.

[16] Mahadevan R, Schilling CH, The effects of alternate optimal solutions in constraint-based genome-scale metabolic models. *Metabolic engineering* 2003, **5**(4):264–76.

[17] Müller AC, Bockmayr A, Fast thermodynamically constrained flux variability analysis. *Bioinformatics (Oxford, England)* 2013, **29**(7):903–9.

[18] Schellenberger J, Que R, Fleming RMT, Thiele I, Orth JD, Feist AM, Zielinski DC, Bordbar A, Lewis NE, Rahmanian S, Kang J, Hyduke DR, Palsson BO, Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox v2.0. *Nature Protocols* 2011, **6**(9):1290–307.

[19] Boele J, Olivier BG, Teusink B, FAME, the Flux Analysis and Modeling Environment. *BMC systems biology* 2012, **6**:8.

[20] Ebrahim A, Lerman JA, Palsson BO, Hyduke DR, COBRApy: COnstraints-Based Reconstruction and Analysis for Python. *BMC systems biology* 2013, **7**:74.

[21] Flamholz A, Noor E, Bar-Even A, Milo R, eQuilibrator–the biochemical thermodynamics calculator. *Nucleic acids research* 2012, **40**(Database issue):D770–5.

[22] Metzler DE, Metzler CM, *Biochemistry: The Chemical Reactions of Living Cells, 2nd Edition*. No. v. 1 in Biochemistry: The Chemical Reactions of Living Cells, San Diego: Academic Press, 2 edition 2001.

[23] Noor E, Haraldsdóttir HS, Milo R, Fleming, RMT, Consistent estimation of Gibbs energy using component contributions. *PLoS Computational Biology* 2013, **9**(7):e1003098.

LIST OF ATTENDEES

(February 13th, 2015)

AMAR Patrick	(pa@lri.fr)
ARRANZ-GIBERT Pol	(polarranzg@hotmail.es)
AUBRY Céline	(celine.aubry@agroparistech.fr)
BADER Joel	(joel.bader@jhu.edu)
BALLET Pascal	(pascal.ballet@univ-brest.fr)
BEN GUEBILA Marouen	(marouen.benguebila@uni.lu)
BEURTON-AIMAR Marie	(beurton@labri.fr)
BIANE Celia	(celia.biane@yahoo.fr)
BOUFFARD Marc	(marc.bouffard@lri.fr)
BOUYIOUKOS Costas	(costas.bouyioukos@issb.genopole.fr)
BUCCHINI François	(francois.bucchini@issb.genopole.fr)
CALLER Ben	(bcaller@gmail.com)
CALZONE Laurence	(Laurence.Calzone@curie.fr)
CARPIO Kristine Joy	(kristine.carpio@dlsu.edu.ph)
CHARVIN Gilles	(gilles.charvin@gmail.com)
CSIKASZ-NAGY Attila	(attila.csikasz-nagy@fmach.it)
DE CALUWÉ Joëlle	(jodecalu@ulb.ac.be)
DE CRAENE Johan	(seahorse378@gmail.com)
DESMEULLES Gireg	(desmeulles@enib.fr)
DOULAZMI Mohamed	(mohamed.doulazmi@upmc.fr)
DUL Zoltan	(zoltan.dul@kcl.ac.uk)
EBENHÖH Oliver	(oliver.ebenhoeh@hhu.de)
FRIANT Sylvie	(s.friant@unistra.fr)
HOREMANS Steff	(steffhoremans@yahoo.com)

KÉPÈS François	(francois.kepes@issb.genopole.fr)
KOUTROUMPAS Konstantinos	(konstantinos.koutroumpas@issb.genopole.fr)
KULIESHOV Igor	(igor.kulieshov@gmail.com)
LE GALL Pascale	(pascale.legall@issb.genopole.fr)
LEPAGE Thibaut	(lepage@issb.genopole.fr)
LE TREUT GUILLAUME	(guillaume.le-treut@issb.genopole.fr)
LONGELIN PÉRON Guillaume	(g0longel@enib.fr)
MADHBOUH Khouloud	(khouloud.madbouh@gmail.com)
MANSY Sheref	(mansy@science.unitn.it)
MAZAT Jean-Pierre	(jean-pierre.mazat@phys-mito.u-bordeaux2.fr)
NOGUE Pierre-Yves	(pierreyves.nogue@gmail.com)
NORRIS Victor	(victor.norris@univ-rouen.fr)
PERES Sabine	(sabine.peres@lri.fr)
RENDALL Alan	(rendall@uni-mainz.de)
RIVIERE Jeremy	(jeremy.riviere@univ-brest.fr)
RÖHL Annika	(annika.roehl@fu-berlin.de)
ROSATI Elise	(elise.rosati@gmail.com)
ROUGNY Adrien	(rougny@lri.fr)
STRECK Adam	(adam.streck@fu-berlin.de)
ZANGHELLINI Juergen	(juergen.zanghellini@boku.ac.at)
ZAWORSKI Julie	(julie.zaworski@issb.genopole.fr)
ZELISZEWSKI Dominique	(dominique.zeliszewski@issb.genopole.fr)
ZHUKOVA Anna	(anna.zhukova@ibgc.cnrs.fr)