Structure and Parameter Estimation for Cell Systems Biology Models

Francisco J. Romero-Campero
ASAP research group, School of Computer Science and IT, University of Nottingham
NG8 1BB, UK
fxc@cs.nott.ac.uk

Hongqing Cao
ASAP research group, School of Computer Science and IT, University of Nottingham
NG8 1BB, UK
hxc@cs.nott.ac.uk

Miguel Cámara
Institute of Infection, Immunity and Inflammation, Center for Biomolecular Sciences, University of Nottingham
NG7 2RD, UK
Miguel.Camara@nottingham.ac.uk

Natalio Krasnogor*
ASAP research group, School of Computer Science and IT, University of Nottingham
NG8 1BB, UK
nxk@cs.nott.ac.uk

ABSTRACT

In this work we present a new methodology for structure and parameter estimation in cell systems biology modelling. Our modelling framework is based on *P systems*, an unconventional computational paradigm that abstracts from the structure and functioning of the living cell. The process of designing models, consisting of both the optimisation of the modular structure and of the stochastic kinetic parameters, is performed using a memetic algorithm. Specifically, we use a nested evolutionary algorithm where the first layer evolves rule structures while the inner layer, implemented also as a genetic algorithm (GA), fine tunes the parameters of the model. Our approach consists of an incremental methodology. Starting from very simple *P system modules* specifying basic molecular interactions, more complicated modules are produced to model more complex molecular systems. These newly found modules are in turn added to the library of available P systems modules so as to be used subsequently to develop more intricate and circuitous cellular models. The effectiveness of the algorithm was tested on three case studies, namely, molecular complexation, enzymatic reactions and autoregulation in transcriptional networks.

Categories and Subject Descriptors

I.6.5 [**Simulation and Modeling**]: Model Development—*Modeling methodologies*; G.1.6 [**Numerical Analysis**]: Optimization; J.3 [**Life and Medical Sciences**]: Biology and genetics—*Systems biology*

General Terms

Algorithms, Experimentation

Keywords

Systems Biology, Synthetic Biology, Modular Biology, P Systems, Memetic Algorithm, Genetic Algorithms

*Corresponding author

GECCO'08 July 12–16, 2008, Atlanta, Georgia, USA.

1. INTRODUCTION

Living cells represent complex biomolecular systems with fully interrelated processes. In order to understand, manipulate or control cellular systems to produce a desirable output, integrate partial knowledge, and to predict their behaviour, the development of *good* models is crucial. A new integrative and interdisciplinary field, *systems biology*, is emerging to study the dynamics of cellular systems as a whole. Systems biology has been presented recently in textbooks and article collections [1, 11, 16]. Systems biology is closely related to the also young and growing research field of *synthetic biology* [3]. The final goal when modelling a cellular system is to obtain the necessary understanding to be able to control its functioning. A deeper knowledge of the organisation and functioning of cellular processes will allow us to engineer our own cellular systems exhibiting a desired behaviour or producing a prefixed output.

The macroscopic continuous and deterministic approach based on ordinary differential equations (ODEs) constitutes the most used methodology in cellular modelling within systems and synthetic biology. Although ODEs have been successfully applied in different systems there are two key assumptions in this approach, namely continuity and determinism that are not always fulfilled. There are many systems where the number of particles of the reacting species are low and the reactions involved are slow. In these systems the previous assumptions are invalid and mesoscopic,

discrete and stochastic approaches are instead more suitable [16]. In this paper we will use a new approach, *P systems*, which integrates this last type of modelling into a computational framework. P systems represent an unconventional computational paradigm [13] that abstracts from the structure and functioning of the living cell. A P system consists of a cell-like membrane structure, with compartments containing multisets of objects which evolve according to given rules. These rules are applied according to an extension of Gillespie's well known Stochastic Simulation Algorithm (SSA) [4] to the multicompartmental structure of P system models [12].

The design of models in systems and synthetic biology is real world problem solving. Therefore, it is a hard process prone to failure where one has to reconsider many times the choice of the model parameters and structure made at different points. The use of computational automated tools can provide the means to optimise both the kinetic parameters and the network topology or rule structure. Nevertheless, the use of computer simulation has been mainly focused on the computation of the corresponding dynamics for a given model parameters and structure. In contrast to this classical approach we have developed a methodology based on *evolutionary algorithms* (EA) that optimises both the kinetic parameters and structure of our models with respect to a given desirable behaviour.

Evolutionary algorithms (EAs)[2] has been developed for solving computation problems in many fields. The approach mimics the process of biological evolution and the mechanisms of natural selection and genetic variation. Suitable codings are used to represent possible solutions to a problem, and the search is guided by using some genetic operators and the principle of *survival of the fittest*. Due to their merits of self-organization, self-learning, intrinsic parallelism and generality, EAs have been successfully applied to economic prediction, optimum control, engineering and scientific computations [5]. In this paper, we propose a memetic algorithm [8] to approach the automatic design of cell systems biology models. Its main idea is to use a nested evolutionary algorithm where the first layer evolves model structures while the inner layer, implemented as a genetic algorithm (GA)[10], acts as a local search for the parameters of the model.

The paper is organised as follows. The next section introduces the modelling methodology and the evolutionary computation methods used in this work. In Section 3 the case studies and the experimental design are presented. Results are discussed in Section 4. Finally, some conclusions and future work are given in Section 5.

2. METHODS

2.1 Modelling Methodology and P systems

In this paper we use a computational, modular and discrete-stochastic modelling approach based on P Systems [13], an emergent branch of Natural Computing introduced by Gh. Paun. More specifically, we use a variant called stochastic P systems developed for the specification and simulation of cellular systems [12].

2.1.1 Stochastic P systems

A stochastic P system is a construct

$$\Pi = (O, L, \mu, M_1, M_2, \ldots, M_n, R_{l_1}, \ldots, R_{l_m})$$

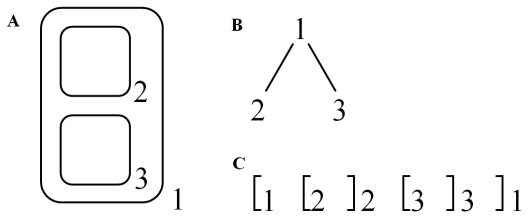

Figure 1: Three possible representations of the same membrane structure consisting of three membranes.

where:

- O is a finite alphabet of *objects* representing molecules.
- $L = \{l_1, \ldots, l_m\}$ is a finite set of *labels* identifying compartment types.
- μ is a *membrane structure* containing $n \geq 1$ membranes defining compartments arranged in a hierarchical manner. Each membrane is identified in a one to one manner with values in $\{1, \ldots, n\}$ and is given a label from L which determines its type. The membrane structure is represented formally, as a rooted tree, where the nodes are called membranes, and the relationship of a membrane being inside another one is represented by the relationship of the node being the descendent of another one. Alternatively, it can be represented using Venn diagrams or pairs of matching square brackets, see Figure 1.
- $M_i = (l_i, w_i)$, for each $1 \leq i \leq n$, is the *initial configuration* of membrane i, with $l_i \in L$ the initial compartment type and w_i the multiset of objects over O initially placed inside the compartment defined by membrane i.
- $R_{l_t} = \{r_1^{l_t}, \ldots, r_{k_{l_t}}^{l_t}\}$, for each $1 \leq t \leq m$, is a finite set of rewriting rules associated with compartments of the type represented by the label $l_t \in L$ and of the following general form:

$$o_1 \, [\, o_2 \,]_l \xrightarrow{c} o_1' \, [\, o_2' \,]_l \tag{1}$$

with o_1, o_2, o_1', o_2' multisets of objects over O (potentially empty) and $l \in L$ a label. These multiset rewriting rules operate on both sides of membranes; a multiset o_1 placed outside membrane l and a multiset o_2 placed inside membrane l are simultaneously replaced by o_1' and o_2', respectively. A stochastic constant c is associated specifically with each rule in order to compute its propensity according to Gillespie's theory of stochastic kinetics [4].

In the original approach in P systems the rules are selected in a non deterministic and maximally parallel manner [13]. Nevertheless, this approach produces a qualitative framework that fails to model quantitative aspects which are key to the functioning of many cellular systems. In order to solve this problem in stochastic P systems the rewriting rules are selected according to an extension of Gillespie's well known Stochastic Simulation Algorithm (SSA) [4] to the multicompartmental structure of P system models [12].

Stochastic P systems have been successfully used in the specification and simulation of cellular systems, for instance signal transduction [12], prokaryotic gene regulation [15] and bacterial colonies [14].

Specifically, in this work we use rules of the following types where the stochastic kinetic constants are only allowed within the given range and with the specified precision.

Table 1: Rules Types

Type	Rule	Constant Range	Precision
0	$[\ X+Y\ ]_l \overset{c}{\rightarrow} [\ Z\ ]_l$	$(10^{-4}, 0.17)$	10^{-2}
1	$[\ Z\ ]_l \overset{c}{\rightarrow} [\ X+Y\ ]_l$	$(10^{-4}, 1)$	10^{-2}
2	$[\ G\ ]_l \overset{c}{\rightarrow} [\ G+R\ ]_l$	$(10^{-4}, 0.2)$	10^{-4}
3	$[\ R\ ]_l \overset{c}{\rightarrow} [\ R+P\ ]_l$	$(10^{-3}, 5\times10^{-2})$	10^{-3}
4	$[\ R\ ]_l \overset{c}{\rightarrow} [\ ]_l$	$(10^{-3}, 10^{-2})$	10^{-3}
5	$[\ P\ ]_l \overset{c}{\rightarrow} [\ ]_l$	$(10^{-5}, 10^{-3})$	10^{-6}
6	$[\ P+G\ ]_l \overset{c}{\rightarrow} [\ P.G\ ]_l$	$(10^{-3}, 0.1)$	10^{-3}
7	$[\ P.G\ ]_l \overset{c}{\rightarrow} [\ P+G\ ]_l$	$(10^{-3}, 0.2)$	10^{-3}
8	$[\ P.G\ ]_l \overset{c}{\rightarrow} [\ P.G+R\ ]_l$	$(10^{-3}, 0.1)$	10^{-3}

Rules of type 0 and 1 represent the formation and dissociation of a molecular complex Z consisting of the individual molecules X and Y. The genetic processes involved in transcription of a gene G into its messenger RNA R and translation of this messenger into the protein product P are specified using rules of type 2 and 3. The degradation of mRNA R and protein P are described by rules of the type 4 and 5. Rules of types 6 and 7 represent the binding and debinding of a protein P to a gene G. Finally, the transcription of a gene occupied by a protein $P.G$ into its messenger RNA R is specified using rule of type 8.

The ranges for the different constants were obtained when possible from lower and upper theoretical bounds [1]. When this was not possible we have chosen empirically a range large enough to find the constants needed in this work.

2.1.2 A Library of basic P system modules

Cellular functions are rarely performed by individual molecular interactions. Most biological functions arise as emergent behaviour from the interactions among modules made up of many molecule species. In this work a *module* is defined as a discrete entity which performs a specific biological function separable from those of other modules. This separation depends on chemical isolation, which can originate from spatial localisation in different compartments or from chemical specificity or time domain separation [6]. These features can be easily represented in P systems using membranes for spatial localisation and rules for chemical specificity.

A P system module is a set of rules with the form in (1) representing molecular interactions which occur repetitively in many cell systems. A module is identified with a name and three sets of variables, V, C and Lab. V represents variables that can be instantiated using objects describing molecules. C represents the stochastic constants associated with each rule. Lab specifies the labels of the compartments involved in the rules. Formally, a module, mo, with variables V, constants C and labels L will be written as $mo(V, C, L)$. More complex modules can be constructed from simple modules by applying set union. Furthermore, the set of rules associated with a membrane in a P system model can be specified in a modular way as the set union of several P system modules.

In what follows, we present the modules included in the initial library of P system modules used in this work:
– Complex formation: Two molecules, X and Y, can collide and stick together according to a stochastic kinetic constant c to produce a complex Z.

$$Com(\{X,Y,Z\},\{c\},\{l\}) = \{[\ X+Y\ ]_l \overset{c}{\rightarrow} [\ Z\ ]_l\} \qquad (2)$$

– Complex dissociation: A molecular complex X can dissociate into its components Y and Z according to a stochastic kinetic constant c.

$$Diss(\{X,Y,Z\},\{c\},\{l\}) = \{[\ X\ ]_l \overset{c}{\rightarrow} [\ Y+Z\ ]_l\} \qquad (3)$$

– Unregulated expression: A gene encoded in the DNA, G, produces the corresponding mRNA, R, which in turn yields a protein P. The mRNA and protein can be degraded by the cell machinery. These processes take place at rates determined by some stochastic constants c_1, c_2, c_3 and c_4.

$$UnReg(\{G,R,P\},\{c_1,c_2,c_3,c_4\},\{l\}) = \\ = \left\{ \begin{array}{l} [\ G\ ]_l \overset{c_1}{\rightarrow} [\ G+R\ ]_l, \\ [\ R\ ]_l \overset{c_2}{\rightarrow} [\ R+P\ ]_l, \\ [\ R\ ]_l \overset{c_3}{\rightarrow} [\ ]_l,\ [\ P\ ]_l \overset{c_4}{\rightarrow} [\ ]_l \end{array} \right\} \qquad (4)$$

– Positive regulated expression: In this case an activator protein Act binds reversibly to the gene G yielding the complex $Act.G$ which turns on the production of the mRNA R. Ultimately, the protein product P is produced from the mRNA. The mRNA and the protein are also degraded in this case. These processes take place at rates determined by some stochastic constants c_1, c_2, c_4, c_5 and c_6.

$$Pos(\{Act,G,R,P\},\{c_1,c_2,c_3,c_4,c_5,c_6\},\{l\}) = \\ = \left\{ \begin{array}{l} [\ Act+G\ ]_l \overset{c_1}{\rightarrow} [\ Act.G\ ]_l, \\ [\ Act.G\ ]_l \overset{c_2}{\rightarrow} [\ Act+G\ ]_l, \\ [\ Act.G\ ]_l \overset{c_3}{\rightarrow} [\ Act.G+R\ ]_l, \\ [\ R\ ]_l \overset{c_4}{\rightarrow} [\ R+P\ ]_l, \\ [\ R\ ]_l \overset{c_5}{\rightarrow} [\ ]_l,\ [\ P\ ]_l \overset{c_6}{\rightarrow} [\ ]_l \end{array} \right\} \qquad (5)$$

– Negative regulated expression: In contrast to the previous case here a repressor protein Rep binds reversibly to the gene G yielding the complex $Rep.G$ which does not produce any mRNA. The binding and debinding of the repressor to the gene take place at a rate determined by some stochastic constants c_1 and c_2.

$$Neg(\{Rep,G\},\{c_1,c_2\},\{l\}) = \\ = \left\{ \begin{array}{l} [\ Rep+G\ ]_l \overset{c_1}{\rightarrow} [\ Rep.G\ ]_l \\ [\ Rep.G\ ]_l \overset{c_2}{\rightarrow} [\ Rep+G\ ]_l \end{array} \right\} \qquad (6)$$

2.2 A Memetic Algorithm for Evolving P system models

We propose a memetic algorithm to evolve P system models for a particular biological signature. Our methodology is a nested evolutionary algorithm where the first layer searches for model *structures* using a genetic algorithm (GA); while the inner layer, also implemented as a GA, acts as a local search for the *parameters* of the model. A detailed flowchart of the algorithm is shown in Figure 2.

2.2.1 Modular Structure Optimisation of P System Models

Encoding: In this work we focus on the design of models of bacterial systems, consequently the membrane structure of all our models consists of a single membrane or compartment. In order to characterise a given P system with a single compartment $\Pi = (O, \{l\}, [\ \ ], M_1, R_{l_1})$ it is sufficient to specify the modules used to obtain the rules $R_{l_1} = \{r_1, \ldots, r_n\}$. Therefore, the stochastic P system model Π can be represented as a vector whose components are the modules describing the set of rules R_{l_1}, $\Pi = (m_1, \ldots, m_n)$.

Each rule is encoded using a structure which specifies the rule type according to Table 1, left hand side (LHS) (reactants), right hand side (RHS) (products) and the stochastic

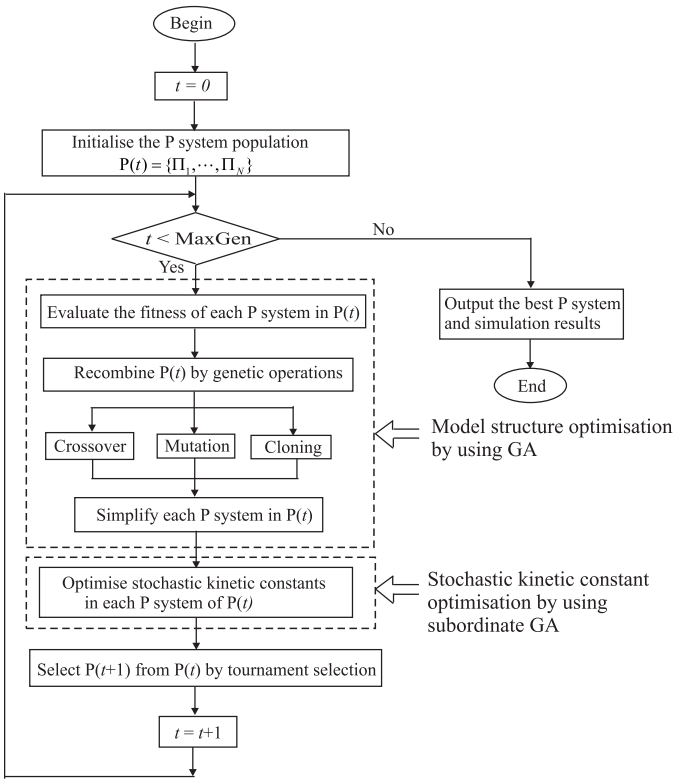

Figure 2: Flowchart of the memetic algorithm for evolving P systems.

kinetic constant. A P system module is then encoded using a structure which specifies the module type, module size (number of rules) and the set of rules included in the module. In Table 2 we associate module types with the elementary modules introduced in the previous subsection.

Table 2: Module Types in the Initial Library of P System Modules

Module name	Module type	Module size
$UnReg$	0	4
Pos	1	6
Neg	2	2
Com	3	1
$Diss$	4	1

For instance, a P system model consisting of the following two modules $UnReg(\{geneA, rnaA, A\}, \{c_1, c_2, c_3, c_4\}, \{l\})$ and $Neg(\{A, geneA\}, \{c_5, c_6\}, \{l\})$ is encoded using the structure in Figure 3.

Fitness evaluation: Given a stochastic P system model, Π, and target time series with N time steps for the evolution of some M specific molecules (e.g. mRNA, protein, metabolite etc) of Π, (O_{ij}^{tar}) where $1 \leq i \leq M$ and $1 \leq j \leq N$, the fitness of the model is calculated using the Root Mean Square Error (RMSE). The provided time series describe the expected behaviour of the target model and therefore they can not be compared directly with a single run of our candidate stochastic P system Π. In order to get an estimation of the expected behaviour of Π, we run a sufficient number of simulations, `MaxRun`, and average them. Figure 4 shows the flowchart of the fitness evaluation process.

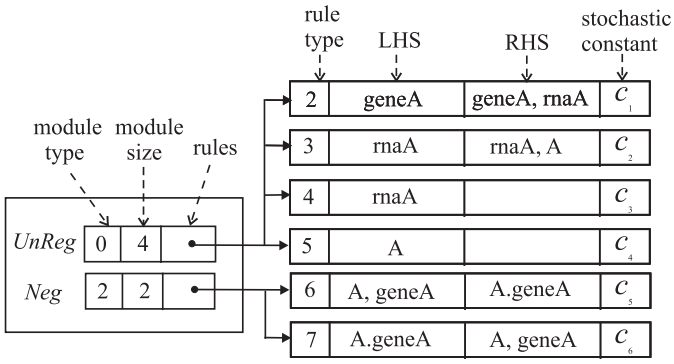

Figure 3: Encoding of a P system model consisting of two modules.

Genetic operators: In the GA used for the optimisation of the modular structure we use *crossover* and *mutation* as the genetic operators.

Crossover can be done by exchanging single modules, *module-exchange crossover*, or by swapping multiple modules between two parents, *one-point crossover*. Consider two parents $\Pi_1 = (m_1^1, \ldots, m_{n_1}^1)$ and $\Pi_2 = (m_1^2, \ldots, m_{n_2}^2)$ with n_1 and n_2 modules respectively.

In the module-exchange crossover, two crossover points, i and j, are randomly selected within Π_1 and Π_2 and then the crossover is performed as follows:

if $m_i^1 \cap m_j^2 = \emptyset$
then swap m_i^1 *and* m_j^2
else swap the stochastic kinetic constants of the common rules within m_i^1 *and* m_j^2
Calculate fitness of both offspring and choose the best one

The one-point crossover is performed by randomly selecting one crossover position from Π_1 and Π_2 and swapping all the modules after the crossover points. We use the valid offspring as the crossover offspring whose number of modules does not exceed the predefined maximum module set size, `MaxMsize`. If both offsprings are valid we choose the one with the best fitness.

The structure mutation is performed by randomly choosing a module and making one of the three following variations: (1) choose randomly a rule within the module and change its stochastic kinetic constant using Gaussian mutation; (2) keep the module type but change some of the objects in the module; (3) modify the module type by adding or deleting some rules.

Finally, before the process of the optimisation of the constants we simplify each P system rule structure by deleting redundant modules and useless modules which cannot be applied during the evolution of the P system model.

2.2.2 Parameter Optimisation of P System Models

As the kinetic constant associated with each rule is used in Gillespie algorithm to compute the probability of applying each rule and the waiting time for the rule to be applied [4], the stochastic constants of a P system model determine its behaviour, and thus it is crucial to optimise them in order to obtain a desirable behaviour. Here we designed a GA [18] to optimise the constants of each candidate P system model with the structure generated during the previous stage of our algorithm.

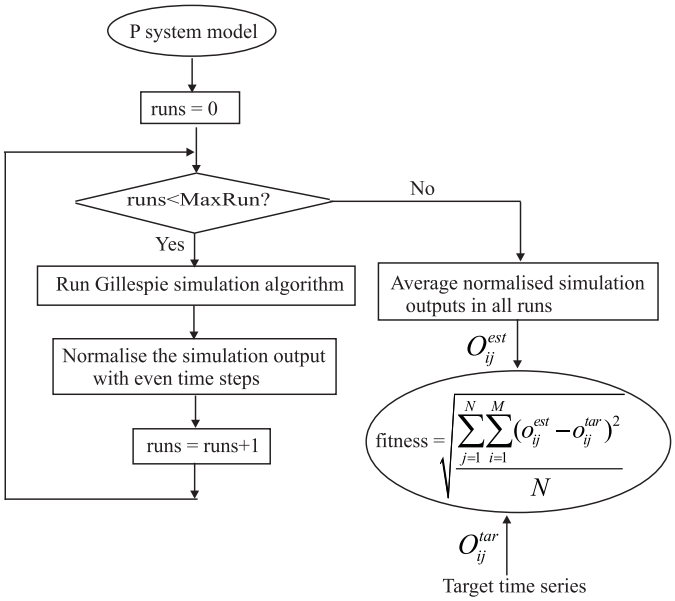

Figure 4: Fitness calculation process.

Encoding: Given a stochastic P system model generated in the previous stage with n modules $\Pi = (m_1, \ldots, m_n)$ first we calculate the total number of different rules, l, in Π by applying set union over the set of rules of the modules $R_\Pi = \cup_{i=1}^{n} m_i = \{r_1, \ldots, r_l\}$. Then we represent each chromosome specifying the constants of Π in the parameter population using an $l-$dimensional row vector $C(\Pi) = (c_1, \ldots, c_l)$ where c_i is the constant associated with r_i for $i = 1, \ldots, l$. Each constant is encoded as a floating number and is generated randomly within the specific range and precision shown in Table 1 during the initialisation of the parameter population.

Fitness evaluation: The fitness of the P system model Π using the constants $C(\Pi) = (c_1, \ldots, c_l)$ is computed according to the process presented in Figure 4.

Genetic operators: During the parameter optimisation we use mutation and crossover. More precisely, we use an adaptation method [7] to alter the mutation rate of individuals r_{mu} using the formula in (7) which uses the fitness of the best constant vector C_{best} and the constant vector of the given individual C. The crossover rate is computed as $r_{cro} = 1 - r_{mu}$.

$$r_{mu} = 0.3\left(1 - \frac{fitness(C_{best})}{fitness(C)}\right) \tag{7}$$

The mutation is performed as follows:

Nmu = 0;
while (Nmu < `MaxMu`*)* {
  Choose randomly one module and one rule within the module;
  Do Gaussian mutation to the constant associated with the rule;
  If the fitness improves then replace the old constant with the new one;
  Nmu ++;}

The crossover is performed using a *multiple-parent crossover operator*. We randomly select $M > 2$ individuals $C_1, \ldots, C_M$ from the parameter population with $C_i = (c_1^i, \ldots, c_l^i)$ for $i = 1, \ldots, M$. Then M coefficients α_i are randomly generated satisfying: (1) $\alpha_i \in (a, b)$ where a and b are parameters of our algorithm such that $a < 0$ and $b > 1$; (2) $\sum_{i=1}^{M} \alpha_i = 1$. Finally, a new vector of constants C_{new} is generated as a nonconvex linear combination of C_i using the previous constants:

$$C_{new} = \sum_{i=1}^{M} \alpha_i C_i \tag{8}$$

If the fitness of new vector of constants C_{new} is better than that of the worst individual C_{worst} in the parameter population then C_{worst} is replaced by C_{new}. This process is iterated a predetermined maximum number of times `MaxXo`.

3. EXPERIMENTS

In this work we use three relatively simple cellular systems as a proof of concept of the feasibility of our methodology.

3.1 Case Studies Definition

The three case studies are *molecular complexation*, *enzymatic reaction* and *autoregulation in transcriptional networks*.

The target time series for all these case studies were generated using PRISM, a probabilistic model checker for formal modelling and analysis of systems that captures random or probabilistic behaviour [9]. In order to obtain the target time series we specified the molecular interactions of each case study as a stochastic process using the PRISM language and then computed the expected behaviour over time checking specific properties specified using temporal logic.

The first two cases, molecular complexation and enzymatic reaction, were used to illustrate the advantage of incorporating newly found modules to the library of available modules. Molecular complexation consists of the formation and dissociation of molecular complexes. This is one of the most important molecular processes as the binding of a molecule to another one can alter the function of the molecular complex which is normally completely different from the functioning of the individual molecules. The second experiment studies enzymatic reactions. During an enzymatic reaction first a molecule called enzyme binds reversibly to specific molecules called substrate. Once bound to the substrate the enzyme performs a change in substrate and finally the complex enzyme-substrate dissociates into the unchanged enzyme and a product. Note that the first step in an enzymatic reaction consists of the molecular complexation of the enzyme and the substrate. In this work we will study how by including a previously discovered P system model for molecular complexation to the library of modules the optimisation of the structure and parameters of a P system model for enzymatic reactions is improved.

The third case study investigates regulation in transcriptional networks. As a proof of concept we start by studying networks consisting of a single gene regulating itself. Although autoregulation is a very simple mechanism, it has been shown to be a highly recurrent pattern in *Escherichia coli* [17]. It consists of a gene whose protein product regulates its own transcription either by repression, *negative autoregulation*, or enhancement, *positive autoregulation*. In this paper we study these two mechanisms and check how

many different designs or P system models our algorithm suggests.

3.2 Parameter Settings and Measures

The parameter settings of our memetic algorithm are listed in Table 3. 50 runs were conducted independently for each case study. All the experiments were performed on a 1024 CPU 2.2GHz gigabit cluster and programmed in C.

Table 3: Parameter Settings of the Memetic Algorithm

GA for structure optimisation	`popsize` = 100 `MaxGen` = 20 `MaxMsize` = 4 `MaxRun` = 50
GA for parameter optimisation	`popsize` = 50 `MaxMu` = 100 `MaxXo` = 100 $M = 8$ a = -0.5 b = 1.5

To evaluate the goodness of the evolved P system model, we use RMSE calculated by following the process shown in Figure 4. The smaller the RMSE, the better the model is.

4. RESULTS AND DISCUSSIONS

In our first case study, the target time series describe reversible molecular complexation of two molecules A and B into the complex C. In PRISM, the reaction forming the complex C was characterised by the constant $c_1^{tar} = 0.048\, molec^{-1} sec^{-1}$, and the dissociation of the complex C into A and B was characterised by $c_2^{tar} = 0.5\, sec^{-1}$. When given the target time series and the library of elementary modules introduced in section 2.1.2, our algorithm always found the same P system model structure which consists of the two modules in (9).

$$\begin{aligned} &RevComp(\{A,B,C\},\{c_1,c_2\},\{l\}) = \\ &= Com(\{A,B,C\},\{c_1\},\{l\}) \cup Diss(\{C,A,B\},\{c_2\},\{l\}) = \\ &= \left\{ \; [\, A+B \,]_l \overset{c_1}{\rightarrow} [\, C \,]_l,\; [\, C \,]_l \overset{c_2}{\rightarrow} [\, A+B \,]_l \; \right\} \end{aligned} \tag{9}$$

The best model produced the following estimation of the constants $c_1^{evo} = 0.048\, molec^{-1} sec^{-1}$ and $c_2^{evo} = 0.48\, sec^{-1}$. The target time series and the behaviour of the best evolved P system model are shown in Figure 5 (top). As it can be seen from the graph, the obtained P system matches the target time series very accurately.

The target time series in the second case study correspond to an enzymatic reaction with a substrate A, enzyme B and product D. The formation and dissociation of the complex enzyme-substrate, C, were characterised in PRISM by the constants $c_1^{tar} = 0.048\, molec^{-1} sec^{-1}$ and $c_2^{tar} = 0.5 sec^{-1}$ whereas the dissociation of the complex into the unchanged enzyme and product was characterised by the constant $c_3^{tar} = 0.25 sec^{-1}$.

In order to find suitable P system models for this case we experimented with two different approaches. In the first approach we used the same elementary library of modules as we did for the first case. In the second approach we added the newly found module in (9) to the library and examined the advantage by comparing with the first approach.

Our experimental results show that using the first approach only in 12 of 50 runs our algorithm found the correct rule structure corresponding with the three molecular interactions forming enzymatic reactions. Whereas using the second approach the correct rule structure consisting of the modules presented in (10) was found in 39 runs. The estimated parameters in the best model found using the second approach are $c_1^{evo} = 0.049\, molec^{-1} sec^{-1}$, $c_2^{evo} = 0.5 sec^{-1}$ and $c_3^{evo} = 0.26 sec^{-1}$, which are closer to the target constants than the ones found using the first approach with

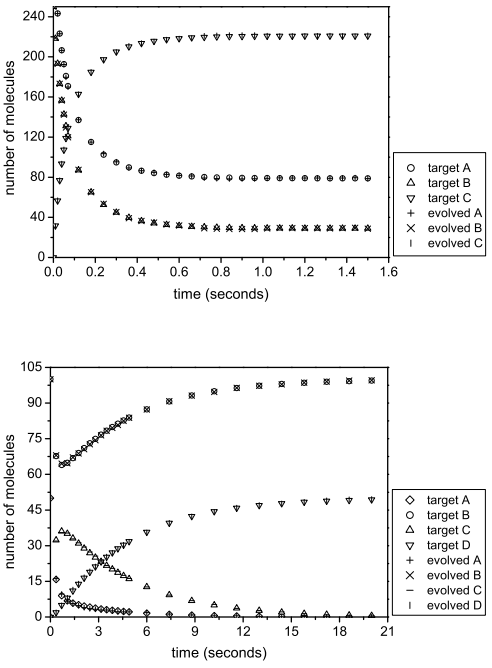

Figure 5: Target time series and the behaviour of the best evolved models for molecular complexation (top) and enzymatic reactions (bottom).

$c_1^{evo} = 0.036\, molec^{-1} sec^{-1}$, $c_2^{evo} = 0.27 sec^{-1}$ and $c_3^{evo} = 0.24 sec^{-1}$. Figure 5 (bottom) depicts the target time series for enzymatic reactions and the behaviour of the best P system model found by using the second approach. It is obvious that all the output molecules A, B, C, D coincide with the target very well.

$$\begin{aligned} &Enz(\{A,B,C,D\},\{c_1,c_2,c_3\},\{l\}) = \\ &RevCom(\{A,B,C\},\{c_1,c_2\},\{l\}) \cup Diss(\{C,B,D\},\{c_3\},\{l\}) = \\ &= \left\{ \begin{array}{l} [\, A+B \,]_l \overset{c_1}{\rightarrow} [\, C \,]_l \\ [\, C \,]_l \overset{c_2}{\rightarrow} [\, A+B \,]_l \\ [\, C \,]_l \overset{c_3}{\rightarrow} [\, B+D \,]_l \end{array} \right\} \end{aligned} \tag{10}$$

In addition, our experimental results showed that by adding the newly found module *RevComp* the mean RMSE got smaller, 2.85, than when using the library of basic modules, 3.8.

In the third case we study transcriptional regulatory networks. More precisely, networks with a single node regulating itself either negatively or positively, namely, negative or positive autoregulation.

Table 4 lists the statistical results obtained in 50 runs using the elementary library introduced in Section 2.1.2. Our algorithm found two possible designs for each case.

Table 4: Statistical Results for Negative and Positive Autoregulation

Case Study	P system Models	Freq.	Mean ± STD
Negative Autoregulation	$\Pi_1 = (UnReg, Neg)$	46	4.11 ± 1.73
	$\Pi_2 = (UnReg, Pos)$	4	4.63 ± 2.91
Positive Autoregulation	$\Pi_3 = (UnReg, Pos)$	30	16.36 ± 3.03
	$\Pi_4 = (UnReg)$	20	20.14 ± 5.13

Table 5: The Target and the Best Models for Negative Autoregulation

Negative Autoregulation				
Design 1		Design 2		Target
Modules	Constants	Modules	Constants	Constants
$UnReg$	0.2 0.03 0.002 $5.23 \cdot 10^{-4}$	$UnReg$	0.2 0.033 0.002 $7.63 \cdot 10^{-4}$	0.13 0.04 0.002 $5.78 \cdot 10^{-4}$
Neg	0.1 0.2	Pos	0.086 0.152 0.002	0.056 0.147
RMSE = 1.703		RMSE = 2.398		

In the case of negative autoregulation, the first design Π_1 was found in 46 runs whereas the second one Π_2 was found only four times. Nevertheless, due to the fact that the mean and standard deviation of the RMSE is comparable in both cases, it shows that both designs are equally plausible. Table 5 shows the best model for each design. The best model for Design 1 consists of two modules, $UnReg$ and Neg, which represents the situation when the binding of the protein completely stops transcription of the gene. This is exactly the rule structure codified in PRISM to generate the target time series. More importantly, by comparing the corresponding constants, it shows that all the estimated parameters in the model are very close to those of the target model. This further demonstrates the high effectiveness of our GA for parameter optimisation. Interestingly, despite the low occurrence frequency, our algorithm can find an alternative design, Design 2, which reproduces the same dynamics as Design 1. Design 2 consists of two modules, $UnReg$ and Pos, which corresponds to the situation when the binding of the protein to the gene does not block transcription completely. This design may be more biologically realistic as it has been reported that the binding of a repressor to a gene normally allows for a leakage in transcription of the repressed gene. Figure 6 depicts the target time series and the behaviour of the best models for the two designs for negative autoregulation with Design 1 (top), Design 2 (bottom). It is notable that both match the target time series very well. The fact that our protocol produces alternative models for a specific biological signature is very encouraging as it could help biologists to design new experiments to discriminate among competing hypotheses (models).

In the case of positive autoregulation, the first design Π_3 was found in 30 runs whereas the second one Π_4 was found 20 times. The mean and standard deviation of the RMSE indicates that the first design is slightly better than the second one. Table 6 shows the best model for each design. The best model for Design 1 consists of two modules, $UnReg$ and Pos, which represents the situation when the protein encoded in the gene directly binds to the gene itself enhancing its own transcription. This is exactly the rule structure codified in PRISM to generate the target time series. For this case study, similarly, our algorithm found an alternative design, Design 2, which reproduces similar dynamics as the target time series. Design 2 only consists of the module $UnReg$, which corresponds to the situation when the protein encoded in the gene does not regulate the transcription rate of the gene. The result suggests that this mechanism can produce similar dynamics as the positive autoregulation by having a high unregulated transcription and a low protein degradation rate which was evidenced by the values of the

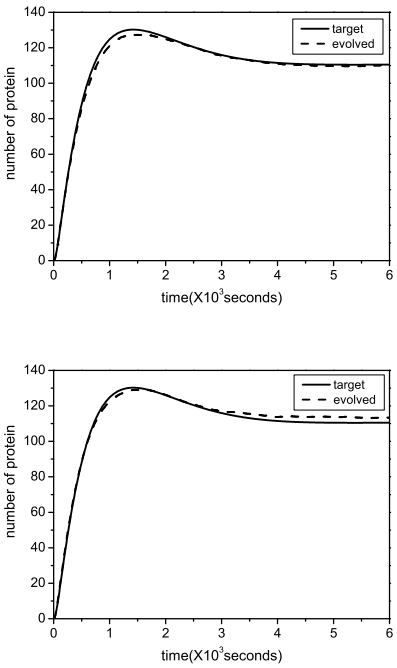

Figure 6: Target time series and the behaviours of the best two possible models for negative autoregulation: Design 1 (top), Design 2 (bottom).

Table 6: The Target and the Best Models for Positive Autoregulation

Positive Autoregulation				
Design 1		Design 2		Target
Modules	Constants	Modules	Constants	Constants
$UnReg$	0.0021 0.006 0.007 $6.3 \cdot 10^{-5}$	$UnReg$	0.0079 0.005 0.002 $3.9 \cdot 10^{-5}$	0.0004 0.016 0.006 10^{-4}
Pos	0.007 0.018 0.031			0.04 0.02 0.014
RMSE = 9.871		RMSE = 13.315		

first constant and the fourth constant for Design 2 listed in Table 6.

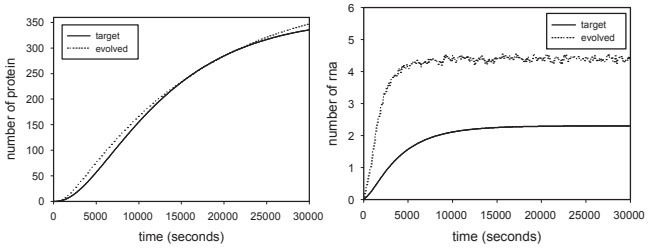

Figure 7: Target time series and the behavior of the best model in Design 1 for positive autoregulation.

Figure 7 shows the target time series and the behaviour of the best model with Design 1 for positive autoregulation.

Note that the dynamics of the protein (left) fits the target time series very accurately. However, as for the rna, it has a similar shape as the target time series but twice the magnitude. Hence in order to improve the simulation result of rna, we changed the weight coefficients of protein and rna in the fitness function from 1:1 to 1:100. Figure 8 illustrates the behaviour of the best model obtained in 50 runs. As it can be seen the dynamics of protein did not change much while the simulation result of rna was improved significantly.

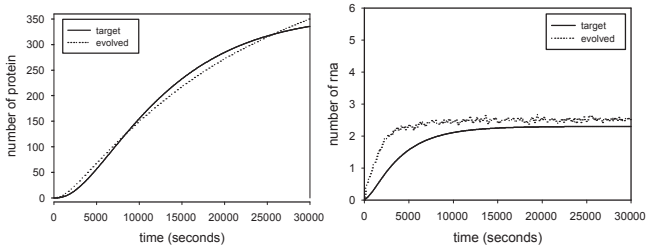

Figure 8: Target time series and the behaviour of the best model for positive autoregulation after changing the fitness function.

5. CONCLUSIONS AND FUTURE WORK

A memetic algorithm has been designed to automatically develop and optimise both the modular structure and parameters of cellular models based on stochastic P systems. Specifically, we use a nested evolutionary algorithm where the first layer evolves model structures while the inner layer implemented as a GA, acts as a local search for the parameters of the model.

The effectiveness of the algorithm was tested by three case studies with incremental model complexity, namely, molecular complexation, enzymatic reactions and autoregulation in transcriptional networks. The experimental results demonstrate that for each case our algorithm was able to find a rule structure and stochastic kinetic constants which reproduce very accurately the target time series. More interestingly, in the case of autoregulation in transcriptional networks our algorithm was able to find more than one equally plausible model (hypothesis). This shows the suitability of our algorithm to find a set of possible designs of cellular systems. Furthermore one of these possible designs could be chosen to be actually engineered in the lab according to its implementation feasibility. This links our research with the emerging field of synthetic biology. Finally, the comparing results for the case of enzymatic reactions by using the elementary library and by adding the newly found module shows the obvious advantage of using the latter approach. This points out the great potential to automatically design more complex and circuitous cellular models in the future by using our algorithm.

Future research lines include the automatic design of more complex regulatory transcriptional networks and the study of eukaryotic cellular systems with relevant compartmentalised structure by using and extending our algorithm.

Acknowledgements

We would like to acknowledge grants EP/E017215/1 and BB/F01855X/1.

6. REFERENCES

[1] U. Alon. *An Introduction to Systems Biology.* Matematical and Computational Biology Series. Chapman & Hall/Crc., 2006.

[2] T. Bäck, U. Hammel, and H.-P. Schwefel. Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation*, 1(1):5–16, 1997.

[3] S. A. Benner and A. M. Sismour. Synthetic biology. *Nature Review, Genetics*, 6:533–543, 2005.

[4] D. T. Gillespie. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.*, 58:35–55, 2007.

[5] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, MA: Addison Welsey, 1989.

[6] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray. From molecular to modular cell biology. *Nature, Impacts*, 402:c47–c52, 1999.

[7] R. Hinterding, Z. Michalewicz, and A. Eiben. Adaptation in evolutionary computation: a survey. In *Proc. 4th International Conference on Evolutionary Computation*, pages 65–69. IEEE Press, 1997.

[8] N. Krasnogor and J. Smith. A tutorial for competent memetic algorithms: model, taxonomy and design issues. *IEEE Transactions on Evolutionary Computation*, 9(5):474– 488, 2005.

[9] M. Z. Kwiatkowska, G. Norman, and D. Parker. Probabilistic model checking in practice: case studies with prism. *SIGMETRICS Performance Evaluation Review*, 32(4):16–21, 2005.

[10] M. Mitchell. *An introduction to genetic algorithm.* MIT Press, Cambridge MA, 1996.

[11] B. O. Palsson. *Systems Biology: Properties of Reconstructed Networks.* Cambridge University Press., 2006.

[12] M. J. Pérez-Jiménez and F. J. Romero-Campero. P systems, a new computational modelling tool for systems biology. *Transactions on Computational Systems Biology VI*, pages 176–197, 2006.

[13] G. Păun. *Membrane Computing: An Introduction.* Springer-Verlag., Berlin, 2002.

[14] F. J. Romero-Campero and M. J. Pérez-Jiménez. A model of the quorum sensing system in vibrio fischeri using p systems. *Artificial Life*, 14(1):95–109, 2008.

[15] F. J. Romero-Campero and M. J. Pérez-Jiménez. Modelling gene expression control using p systems: The lac operon, a case study. *BioSystems*, 91 (3):438–457, 2008.

[16] Z. Szallasi, J. Stelling, and V. Periwal. *System modeling in cellular biology.* MIT press, 2006.

[17] D. Thieffry, A. Huerta, E. Perez-Rueda, and J. Collado-Vides. From specific gene regulation to genomic networks: A global analysis of transcriptional regulation in escherichia coli. *BioEssays*, 20(5):433–440, 1998.

[18] J. Yu, H. Cao, Y. He. A new tree structure code for equivalent circuit and evolutionary estimation of parameters. *Chemometrics and intelligent laboratory systems*, 85:27–39, 2007.