Conventional Verification for Unconventional Computing: a Genetic XOR Gate Example

Savas Konur[1], Marian Gheorghe[1], Ciprian Dragomir[1], Florentin Ipate[2], Natalio Krasnogor[3]

[1] Department of Computer Science, University of Sheffield, UK
{s.konur,m.gheorghe,c.dragomir}@sheffield.ac.uk
[2] Department of Computer Science, University of Bucharest
florentin.ipate@ifsoft.ro
[3] School of Computing Science, Newcastle University, UK
natalio.krasnogor@newcastle.ac.uk

Summary. As unconventional computation matures and non-standard programming frameworks are demonstrated, the need for formal verification will become more prevalent. This is so because "programming" in unconventional substrates is difficult. In this paper we show how conventional verification tools can be used to verify unconventional programs implementing a logical XOR gate.

1 Introduction

Unconventional computing, with many aspects including implementations in vivo, vitro and silico, models and methods, programming paradigms and tools, is a rapidly growing research area with results, promises and huge hope in building new computational devices and tools for solving better or/and faster increasingly complex problems than current machines, models and tools, which either produce inefficienct results or are just unable to solve them.

One specific class of models and experiments related to unconventional computing, often called natural computing, is inspired by natural processes occurring in biology or produces in vitro (DNA strands) or in vivo (bacteria) experiments simulating different computational devices. A thorough account of the developments in the area can be found in [1], but we also mention some specific demonstrations of unconventional computing using liposomes [2], programmable polymers [3] and photochromic molecules [4].

XOR gate is a classic computer science concept with various unconventional computing incarnations. More recently some implementations have been provided [5, 6] and solutions using synthetic biology computational tools have been proposed [7].

In this paper we aim to reconsider this problem and to provide a set of unconventional computing models based on the P systems computational paradigm [8]. Here, we consider *stochastic P systems* [9] and *kernel P systems* [10] as representative classes of such models. These models are associated with verification methods using model checking approaches.

The key contributions of the paper are: the *introduction of a set of unconventional models based on P systems*, which naturally describe the genetic XOR gate problem, and the *use of some model checkers for verifying properties of the models.* This approach is complementary to the previous investigations and highlights new perspectives for investigating these systems.

2 Stochastic and Non-deterministic P Systems: Basic Concepts and Tools

Membrane computing [8] is a branch of natural computing inspired by the hierarchical structure of the living cell. The central model, called *P system*, consists of a membrane structure, the regions of which contain rewriting rules operating on multisets of objects [8]. The P system *evolves* by repeatedly applying rules, mimicking chemical reactions and transportation across membranes or cellular division or death processes, and halts when no more rules can be applied. The most recent developments in this field are reported in [11].

The closeness of this model to the biology makes it highly suited as a specification vehicle for representing biological systems, especially (multi-)cellular systems and molecular interactions taking place in different locations of living cells [12]. Different simple molecular interactions or more complex gene expressions, compartment translocation, as well as cell division and death are specified using multiset rewriting or communication rules, and compartment division or dissolution rules. In the case of stochastic P systems, constants are associated with rules in order to compute their probabilities and time needed to be applied, respectively, according to the Gillespie algorithm [13]. This approach is based on a Monte Carlo algorithm for stochastic simulation of molecular interactions taking place inside a single volume or across multiple compartments.

Definition 1. *A **stochastic P system (SP system)** is a model consisting of a tissue P system with a stochastic semantics [13]:*

$$SP = (O, L, \mu, M_1, \ldots, M_n, R_1, \ldots, R_n) \tag{1}$$

where O is a finite set of objects, called *alphabet,* denoting the entities involved in the system; L is a finite set of *labels* naming compartments; μ is a *membrane structure* composed of $n \geq 1$ membranes defining the regions or compartments of the system and their connections, forming an *arbitrary graph*; $M_i = (l_i, w_i)$, $1 \leq i \leq n$, is the *initial configuration* of the compartment or region defined by the membrane i, where $l_i \in L$ is the label of the compartment and $w_i \in O^*$

is a finite *initial multiset of objects*; $R_i = \{r_1^i, \ldots, r_{m_i}^i\}$, $1 \leq i \leq n$, is a set of *multiset rewriting rules*, of the form: $r_k^i : [\ x \xrightarrow{c_k} y\ ]_{l_i}$, where x and y are multisets of objects (y might be empty) over O, representing the molecular species consumed and produced in the corresponding molecular interaction occurring in the compartment labelled l_i. An application of a rule of this form changes the content of the membrane with label l_i by replacing the multiset x with y. The stochastic constant c_k is used by the Gillespie algorithm [14] in order to compute the probabilities associated with the rules [13].

The model has been used as a basis for a specification language [13, 15] and applied, among others, in unconventional computing using liposomes [2] and specifying a synthetic biology pulse generator [16].

The model also includes communication rules, but these are not discussed in this paper as the system we deal with consists of one single compartment without communication rules. In this case the label of the compartment will be also dropped.

Certain systems can be modelled with P systems which do not require probabilistic features. In [12] some types of P systems without probabilities are presented. These variants are utilised for specifying biological systems. **Kernel P systems (kP systems)** have been introduced as a unifying framework allowing to express within the same formalism many classes of non-deterministic P systems [10, 17]. In this paper we use this class of systems only for very limited purposes, obtaining them directly from the stochastic ones and making use of some tools associated with them. The kP systems derived from SP systems use the same alphabet and rules without kinetic constants. In general, each kP system model has explicitly defined execution strategies for its components. In this paper the execution strategy consists of executing one single rule per step, non-deterministically chosen from the set of rules that can be applied.

The **Infobiotics Workbench** (IBW) tool [16, 15] has been built for modelling and prototyping biological systems exhibiting molecular interactions. It allows to define such systems using the above mentioned formalism, SP systems, providing support for the simulation, verification, analysis and optimisation of these models. The experiments to be discussed later have been performed using IBW. The XOR gate will be modelled using SP systems and then simulated and formally verified. The formal verification is performed with third party tools, PRISM [18] and MC2 [19] (integrated in this framework). The corresponding kP model will be verified using SPIN [20].

PRISM [18] is a very popular and widely used probabilistic model checker. It allows probabilistic properties, supporting PCTL [21] (a probabilistic extension of temporal logic) and Continuous Stochastic Logic (CSL [22]). Both languages make use of special operators to express quantitative information which is useful for a precise, fine grain analysis. The property languages also allow describing *reward*-based properties to express quantitative expressions. PRISM suffers from the same problem exhibited by all model checkers, namely state space explosion

and consequently cannot cope with very large state spaces. This is overcome by an alternative model checking approach, *statistical model checking*.

MC2 [19] is a *statistical* model checker, where properties are analysed against a finite set of simulation traces using statistical methods, e.g. Monte Carlo. Unlike symbolic and numerical methods, e.g. those employed in PRISM, statistical model checkers do not analyse the system *exhaustively*, which increases the performance significantly. In MC2, properties are expressed using PLTL_c [19], a probabilistic extension of LTL with constraints. PLTL_c allows properties with some functions returning *maximum/minimum* values of a species and "*derivative* of the concentration of species at each time point" [19].

The **kP Workbench** (kPW) tool [17] has been built to support kP systems formalism, allowing simulation and formal verification. It uses a specific language, based on kP systems, kP-lingua, allowing to specify non-deterministic rule-based systems. The formal verification is performed using a model checking approach based on SPIN, which is incorporated into the framework. The models written in kP-lingua are automatically translated into SPIN. The non-deterministic version of the XOR gate model will be specified using kP systems and the formal verification will be provided in SPIN.

SPIN [20] is a widely used model checking tool with many applications in concurrent and distributed systems verification. A high level modelling language, PROMELA, suitable for describing concurrent processes and interprocess communication, is at the core of this tool. SPIN provides complete support for *Linear-time Temporal Logic* (LTL) and *on the fly* verification procedures which avoid the necessity to generate the global state space prior to performing a search.

3 XOR Gate and Unconventional Models

In this section we consider a genetic XOR logic gate. This has been designed in various papers, including [5, 6]. The construction used in this paper is taken from [7] where it is defined in GEC, a language for synthetic biology. The gate expresses the green fluorescent protein (GFP) if either of aTc or IPTG molecules are present, but not both. Figure 1 illustrates the genetic construction and the corresponding network.

The XOR device comprises two mechanisms. Each mechanism leads to the production of GFP, when it is activated; but two mechanisms cannot be activated at the same time. Namely, while one is active, the other one is inhibited by some protein.

In this system, the transcription factors LacI and TetR are expressed by a gene controlled by the same promoter. The LacI and TetR proteins work in the opposite way. LacI represses the first mechanism, but promotes the other one. On the other hand, TetR promotes the first mechanism, while inhibiting the second. In other words, both proteins serve as inhibitor and promoter in a complementary fashion. In each mechanism, while one protein is an inhibitor, the other one is promoter. When either of the proteins works as an inhibitor, it binds to the corresponding

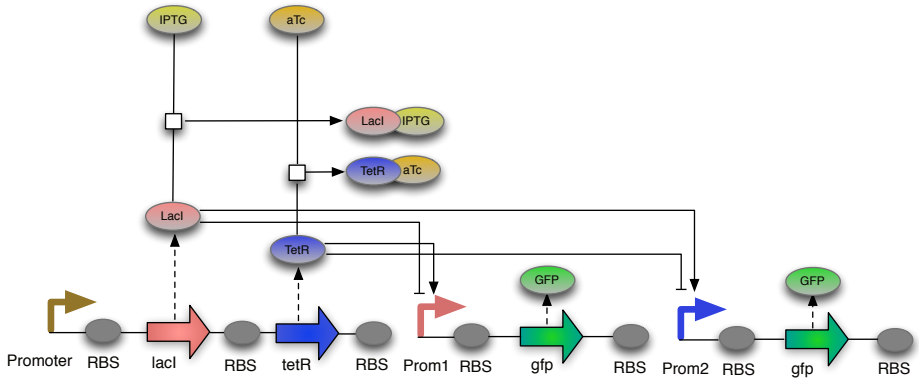


Fig. 1: The genetic parts and design of the XOR gate (redrawn from [7]).

promoter which upregulates the expression of GFP. The XOR device receives two input signals: aTc and IPTG. The aTc and IPTG signals bind to TetR and LacI, respectively, to prevent them interacting with the promoters producing GFP.

Two mechanisms together ensure that the production of GFP will be low when both input signals are set to very low or very high concentrations at the same time. In the former case, LacI and TetR will be produced in abundance, which will then repress the GFP expression. In the latter case, the LacI and TetR concentration will be very low, which is not sufficient to express GFP. On the other hand, if one signal is set to high and the other one is set to low, the device will produce high amount of GFP, hence will act as a Boolean XOR gate.

3.1 Stochastic model

The stochastic model comprises a single compartment with initial concentrations of aTc and IPTG molecules and a set of SP system rules, which govern the kinetic and stochastic behaviour of the system. The initial values are those illustrated in Figures 3. The rewriting rules and the kinetic constants, provided in Table 1, describe the model provided in [7]. A gene controlled by the same promoter expresses LacI and TetR (rules r_1 to r_3). Rules r_{7a} and r_{15a} describe the inhibition of the two mechanisms leading to the GFP production by binding to the promoters that upregulates the production process; r_{7b} and r_{15b} define the debinding process. The activation of the first mechanism by the transcription factor TetR binding to the promoter and the activation of the second mechanism by LacI, are modelled by rules r_{9a}, r_{10}, r_{11} and r_{13a}, r_{14}, r_{17}, respectively. Rules r_{9b} and r_{13b} are debinding reactions. Rules r_4 and r_5 define the binding process involving LacI and IPTG and TetR and aTc, respectively. The degradation process of various molecular species is defined by rules r_{18} to r_{23}.

Table 1: XOR reaction rules.

Rule		Stochastic constant
r_1 :	gene_LacI_TetR $\stackrel{k_1}{\rightarrow}$ gene_LacI_TetR + protein_LacI_TetR	$k_1 = 0.12$
r_2 :	protein_LacI_TetR $\stackrel{k_2}{\rightarrow}$ protein_LacI_TetR + LacI	$k_2 = 0.1$
r_3 :	protein_LacI_TetR $\stackrel{k_3}{\rightarrow}$ protein_LacI_TetR + TetR	$k_3 = 0.1$
r_4 :	LacI + IPTG $\stackrel{k_4}{\rightarrow}$ LacI-IPTG	$k_4 = 1.0$
r_5 :	TetR + aTc $\stackrel{k_5}{\rightarrow}$ TetR-aTc	$k_5 = 1.0$
r_6 :	gene_GFP1 $\stackrel{k_6}{\rightarrow}$ gene_GFP1 + protein_GFP1	$k_6 = 0$
r_{7a} :	gene_GFP1 + LacI $\stackrel{k_{7a}}{\rightarrow}$ gene_GFP1-LacI	$k_{7a} = 1.0$
r_{7b} :	gene_GFP1-LacI $\stackrel{k_{7b}}{\rightarrow}$ gene_GFP1 + LacI	$k_{7b} = 0.01$
r_8 :	gene_GFP1-LacI $\stackrel{k_8}{\rightarrow}$ gene_GFP1-LacI + protein_GFP1	$k_8 = 0$
r_{9a} :	gene_GFP1 + TetR $\stackrel{k_{9a}}{\rightarrow}$ gene_GFP1-TetR	$k_{9a} = 1.0$
r_{9b} :	gene_GFP1-TetR $\stackrel{k_{9b}}{\rightarrow}$ gene_GFP1 + TetR	$k_{9a} = 0.5$
r_{10} :	gene_GFP1-TetR $\stackrel{k_{10}}{\rightarrow}$ gene_GFP1-TetR + protein_GFP1	$k_{10} = 0.1$
r_{11} :	protein_GFP1 $\stackrel{k_{11}}{\rightarrow}$ protein_GFP1 + GFP	$k_{11} = 0.1$
$r_{12:}$	gene_GFP2 $\stackrel{k_12}{\rightarrow}$ gene_GFP2 + protein_GFP2	$k_{12} = 0$
r_{13a} :	gene_GFP2 + LacI $\stackrel{k_{13a}}{\rightarrow}$ gene_GFP2-LacI	$k_{13a} = 1.0$
r_{13b} :	gene_GFP2-LacI $\stackrel{k_{13b}}{\rightarrow}$ gene_GFP2 + LacI	$k_{13b} = 0.5$
r_{14} :	gene_GFP2-LacI $\stackrel{k_{14}}{\rightarrow}$ gene_GFP2-LacI + protein_GFP2	$k_{14} = 0.1$
r_{15a} :	gene_GFP2 + TetR $\stackrel{k_{15a}}{\rightarrow}$ gene_GFP2-TetR	$k_{15a} = 1.0$
r_{15b} :	gene_GFP2-TetR $\stackrel{k_{15b}}{\rightarrow}$ gene_GFP2 + TetR	$k_{15b} = 0.01$
r_{16} :	gene_GFP2-TetR $\stackrel{k_{16}}{\rightarrow}$ gene_GFP2-TetR + protein_GFP2	$k_{16} = 0.0$
r_{17} :	protein_GFP2 $\stackrel{k_{17}}{\rightarrow}$ protein_GFP2 + GFP	$k_{18} = 0.1$
r_{18} :	GFP $\stackrel{k_{18}}{\rightarrow}$	$k_{18} = 0.01$
r_{19} :	LacI $\stackrel{k_{19}}{\rightarrow}$	$k_{19} = 0.01$
r_{20} :	TetR $\stackrel{k_{20}}{\rightarrow}$	$k_{20} = 0.01$
r_{21} :	protein_GFP1 $\stackrel{k_{21}}{\rightarrow}$	$k_{21} = 0.001$
r_{22} :	protein_GFP2 $\stackrel{k_{22}}{\rightarrow}$	$k_{22} = 0.001$
r_{23} :	protein_LacI_TetR $\stackrel{k_{23}}{\rightarrow}$	$k_{23} = 0.001$

Given certain initial values for aTc and IPTG, different output values are obtained for the GFP products, as shown above.

3.2 Non-deterministic model

The rules of the non-deterministic model are obtained directly from the set of SP system rules given in Table 1, by removing the kinetic constants. Some of the rules that do not contribute to the model, with kinetic constants equal to 0 (r_6, r_8, r_{12}, r_{16}), are completely removed. The initial values are kept the same as in the stochastic case. The rules are executed in a non-deterministic manner, as

described earlier in this paper. This non-deterministic model allows to describe all chains of reactions, observe various interactions between species and determine various dependencies between molecules. It will be used in this respect as the basic model for qualitative analysis. As in this case we are interested in an efficient behaviour of the system, some simplifications will be made to the model, whereby the number of molecules will be bounded.

4 Experiments

In this section, we will provide a computational analysis to infer the system dynamics of the genetic XOR gate. This approach complements previous in vitro or in silico implementations of this unconventional computational problem [5, 6, 7] with a set of qualitative and quantitative properties and results. The stochastic model introduced in Section 3.1 and non-deterministic model from Section 3.2 will be used as specifications for the experiments that follow. We note that the complete model and experimental results of the XOR gate can be accessed at[4].

4.1 Non-deterministic Model

The non-deterministic model, discussed in Section 3.2 and obtained from the SP system described in Table 1, will form the basis for translation into Spin.

Model checking results.

The experiments made and reported in this section refer to relationships between species occurring on various reaction pathways. First we verify generic relationships between species. The property

"The GFP is preceded by the production of at least one of LacI or TetR"

is formally expressed as

$$\mathrm{F}\ (\mathrm{GFP} > 0)\ \rightarrow\ \neg((\mathrm{LasR} = 0\ \wedge\ \mathrm{TetR} = 0)\ \mathrm{U}\ \mathrm{GFP} > 0),$$

and the result of this property is true.

We cannot make any direct connections between the signal molecules aTc and IPTG, and the GFP produced, as the system is non-deterministic and any combination of the signal molecules may lead to GFP. However, we can be more specific with respect to the above relationships and refer to the production of a transcription factor and its role as a repressor. More specifically, we verify the property

"When there is no TetR in the system and the LacI represses gene_GFP1 then GFP is produced only by the activation of gene_GFP2"

[4] `http://www.dcs.shef.ac.uk/~konur/models/xor`

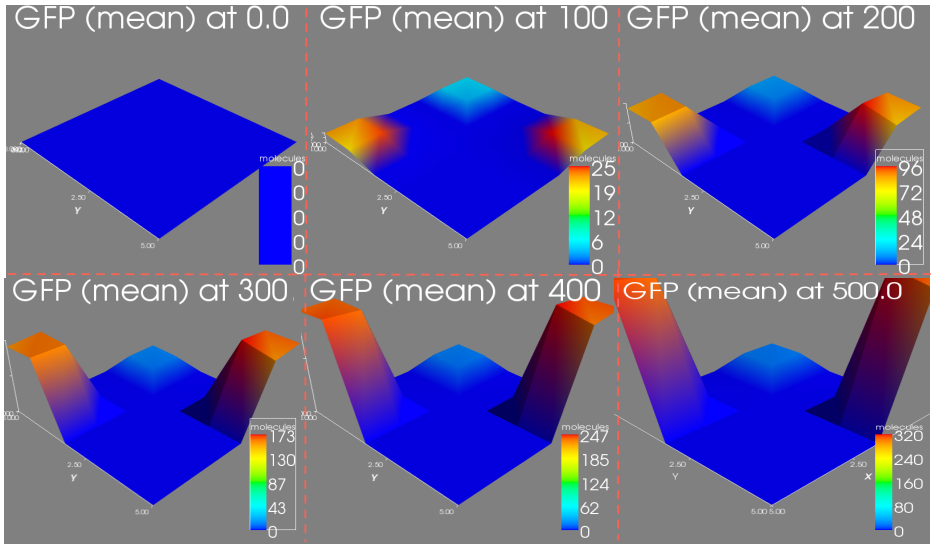


Fig. 2: Simulation of the stochastic model.

which is formally expressed as

$$\mathrm{F}\ (\mathrm{GFP} > 0 \wedge\ \text{gene\_GFP1-LacI} > 0)\ \rightarrow\ \neg((\mathrm{TetR} = 0\ \ \wedge\ \text{gene\_GFP2-LacI} = 0)\ \mathrm{U}\ (\mathrm{GFP} > 0\ \wedge\ \text{gene\_GFP1-LacI} > 0)).$$

The result of this property is true. We can formulate a similar property for TetR.

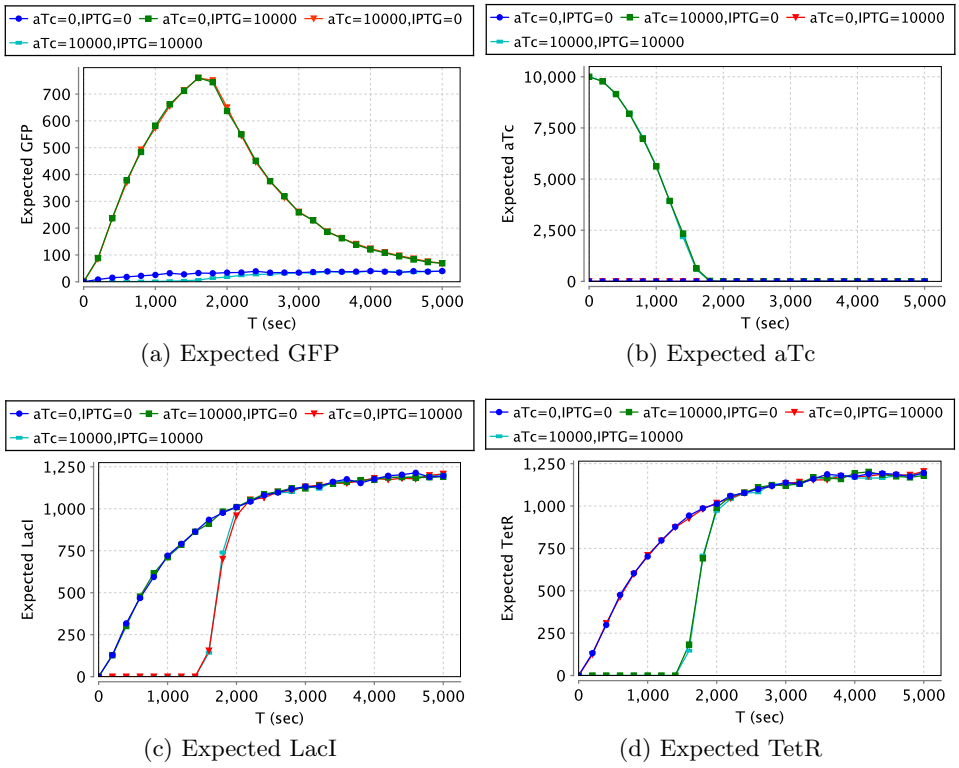
4.2 Stochastic Model

For the stochastic analysis, we have constructed a system model based on SP systems, the modelling language of the Ibw system, using the set of rules discussed in Section 3.1. Below, we summarise some of the experiments that we have carried out using the computational tools integrated into Ibw.

Simulation results.

Figure 2 illustrates the simulation results of the XOR system, performed using the Ibw's Mcss tool, a simulator for multi-compartment SP system models [9]. Ibw provides a GUI to view the simulation results in various formats, e.g. time series, bars, histograms and 3D heat-map animations.

Figure 2 comprises the screen shots of the 3D animation at different time instants. At the top and bottom corners of the lattice both input signals (i.e. aTc



(a) Expected GFP (b) Expected aTc

(c) Expected LacI (d) Expected TetR

Fig. 3: Expected amount of some species based on different initial amounts of aTc and IPTG.

and IPTG) are simultaneously set to very low (i.e. 0) and very high concentrations (i.e. 10000), respectively. Meanwhile, at the left and right corners, one signal is set to very high while the other one is set to very low. As illustrated in the figure, only left and right corners yield a sharp increase in the GFP concentration, ensuring that the designed circuit shows an XOR gate behaviour.

Model checking results.

As discussed above, IBW also permits formal verification of a system using model checking techniques. Since the SP systems allow modelling stochastic models, IBW uses probabilistic model checking tools, currently PRISM and MC2.

Prism results:

We first analyse the amounts of different species over time with four combinations of inputs. The informal property

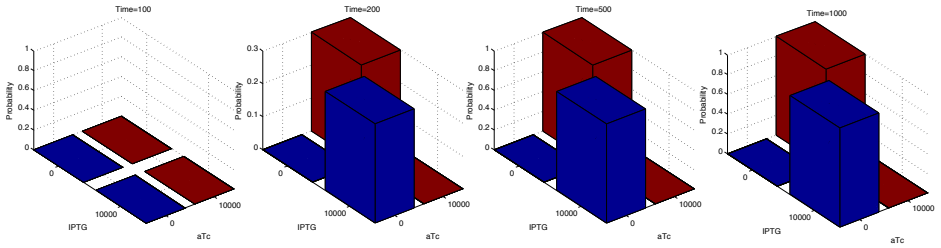


Fig. 4: Probability that GFP exceeds the threshold.

"What is the expected concentration of X at the time instant t?"

is formally expressed as a reward-based formula,

$$\mathrm{R}\{\text{“}X\text{”}\}_{=?}\left[\mathrm{I}=t\right].$$

Figure 3 illustrates the expected amounts of GFP, aTc, LacI and TetR. As shown in Figure 3a, the input combinations aTc=0 – IPTG=10000 and aTc=10000 – IPTG=0 result in a sharp increase in the GFP concentration, whereas the combinations aTc=0 – IPTG=0 and aTc=10000 – IPTG=10000 cause the GFP concentration to stay in low levels, confirming the behaviour of the XOR gate.

Figure 3b shows that if the aTc concentration level is initially set to high, the concentration reduces until it becomes 0. As can be seen in Figure 3d, aTc suppresses the TetR protein by binding to it. After aTc molecules are totally consumed, the TetR concentration starts increasing. We can observe a similar behaviour to Figure 3b, when the IPTG concentration is set to high. IPTG molecules suppress the LacI protein as shown in Figure 3c. These results are inline with the system behaviour, described in Section 3.1.

We now measure the likelihood that the GFP concentration exceeds a certain threshold in any input combination. The property

"What is the probability that GFP *exceeds Thr within t seconds?"*

is formally expressed as

$$\mathrm{P}_{=?}\left[\mathrm{F}^{\leq t}\ \mathrm{GFP} > Thr\right].$$

Figure 4 illustrates the probability values calculated for a threshold value of 100 over different time instants. Clearly, it is almost certainly that GFP exceeds the threshold value for the input combinations aTc=0 – IPTG=10000 and aTc=10000 – IPTG=0. This confirms the desired behaviour.

We now consider a more complex property. Assume that GFP_{ij} (where $i, j \in \{0, 1\}$ represents the state of aTc and IPTG, respectively) denotes the GFP concentration for different input combinations. Namely, if i=0 (resp. j=0), then aTc=0 (resp. IPTG=0), and if i=1 (resp. j=1), then aTc=10000 (resp. IPTG=10000). Then, the property

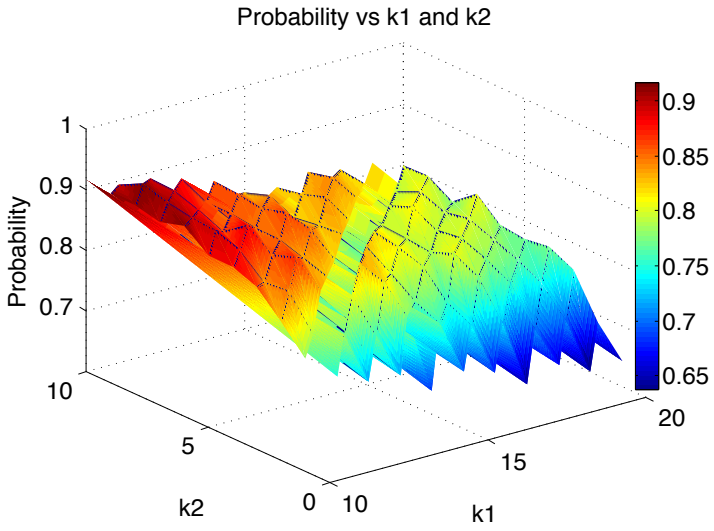


Fig. 5: Parametrised probability formula.

What is the probability that GFP_{01} *and* GFP_{10} *are at least* k_1 *times more than* GFP_{00} *and* GFP_{11}*, and* GFP_{01} *is within the range of* $\mathrm{GFP}_{10} \pm \frac{k_2}{10}$

is formally specified as

$$\begin{aligned} \mathrm{P}_{=?}[\mathrm{F}^{\leq t} \;\; & \mathrm{GFP}_{01} \geq k_1 * \mathrm{GFP}_{00} \;\wedge\; \mathrm{GFP}_{01} \geq k_1 * \mathrm{GFP}_{11} \wedge \\ & \mathrm{GFP}_{10} \geq k_1 * \mathrm{GFP}_{00} \;\wedge\; \mathrm{GFP}_{10} \geq k_1 * \mathrm{GFP}_{11} \wedge \\ & \mathrm{GFP}_{01} \geq (1 - \tfrac{k_2}{10}) * \mathrm{GFP}_{10} \;\wedge\; \mathrm{GFP}_{01} \leq (1 + \tfrac{k_2}{10}) * \mathrm{GFP}_{10}]. \end{aligned}$$

Figure 5 shows the plot based on different k_1 and k_2 values. As expected, the probability becomes higher when k_1 is lower and k_2 is higher, because the formula becomes less strict.

MC2 results:

We now consider a property describing the behaviour in Figure 3a. We want to query

"What is the probability that GFP_{10} *reaches a concentration level of at least* l_1 *times more than the maximum concentrations of* GFP_{00} *and* GFP_{11}*; the concentration then starts decreasing and reduces until it becomes one* l_2 *ration of its maximum level."*

This formula is expressed in PLTL_c as follows:

$$\mathrm{P}_{=?}[\mathrm{F}\ ([\ \mathrm{GFP}_{10}] \geq l_1 * max[\mathrm{GFP}_{00}]\ \wedge\ [\mathrm{GFP}_{10}] \geq l_1 * max[\mathrm{GFP}_{11}] \wedge \\ (\mathrm{F}\ (d[\mathrm{GFP}_{10}] \leq 0\ \wedge\ (\mathrm{F}\ [\mathrm{GFP}_{10}] \leq max[\mathrm{GFP}_{10}]/l_2))))].$$

We have analysed the property for $l_1 = l_2 = 5$, and the probability value returned is 1.0, confirming the behaviour in Figure 3a. We can verify the same property for GFP_{01}, which returns the same result.

5 Conclusions

In this paper we have presented a stochastic P systems model and a non-deterministic one for specifying and studying the behaviour of a genetic XOR gate. These two models are formally analysed using model checking methods revealing qualitative aspects, like expected chain of reactions and dependencies of various species, as well as the quantitative aspects regarding the concentration of certain products with respect to the amount of signal molecules, time to reach certain concentration of molecules or comparisons between maximum concentration achieved for certain species. Our approach is orthogonal to many other unconventional computational investigations or implementations of genetic Boolean gates.

In this line of research, we aim to expand to some other unconventional models, starting with the two genetic XOR gate approaches already mentioned in [5, 6]. We aim also to clarify better the role of various model checkers and types of properties with respect to various systems.

In [23] an interesting prediction and programmability problem for non-DNA molecular self-assembly using porphyrin tiles is investigated. In one of the investigated cases, the self-assembly process is defined as a simple two state probabilistic automaton. The diagonals of a lattice are written with red symbols in one state and blue symbols in the other state, with a small error. This model can be directly represented in Prism and properties regarding the distribution of the two colours on each diagonal or across the lattice are verified. In a forthcoming paper we will be investigating in more details this problem.

Acknowledgements. SK and MG acknowledge EPSRC (EP/I031812/1) support; NK's work is supported by EPSRC (EP/I031642/1, EP/J004111/1, EP/L001489/1). MG and FI are partially supported by CNCS UEFISCDI (PN-II-ID-PCE-2011-3-0688). CD acknowledges an EPSRC studentship.

References

1. Rozenberg, G., Bäck, T., Kok, J.N., eds.: Handbook of Natural Computing. Springer (2012)
2. Smaldon, J., Romero-Campero, F.J., Fernandez Trillo, F., Gheorghe, M., Alexander, C., Krasnogor, N.: A computational study of liposome logic: Towards cellular computing from the bottom up. Systems and Synthetic Biology **4**(3) (2010) 157 – 179

3. Pasparakis, G., Vamvakaki, M., Krasnogor, N., Alexander, C.: Diol-boronic acid complexes integrated by responsive polymers - a route to chemical sensing and logic operations. Soft Matter **4**(20) (2009) 3839 – 3841
4. Chaplin, J.C., Russell, N.A., Krasnogor, N.: Implementing conventional logic unconventionally: Photochromic molecular populations as registers and logic gates. Biosystems **109**(1) (2012) 35 – 51
5. Tamsir, A., Tabor, J.J., Voigt, C.A.: Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'. Nature **469**(7329) (2011) 212–215
6. Regot, S., Macia, J., Conde, N., Furukawa, K., Kjellen, J., Peeters, T., Hohmann, S., de Nadal, E., Posas, F., Sole, R.: Distributed biological computation with multicellular engineered networks. Nature **469**(7329) (2011) 207–211
7. Beal, J., Phillips, A., Densmore, D., Cai, Y.: High-level programming languages for biomolecular systems. In: Design and Analysis of Biomolecular Circuits. Springer New York (2011) 225–252
8. Păun, G.: Computing with membranes. Journal of Computer and System Sciences **61**(1) (2000) 108–143
9. Romero-Campero, F.J., Twycross, J., Camara, M., Bennett, M., Gheorghe, M., Krasnogor, N.: Modular assembly of cell systems biology models using P systems. International Journal of Foundations of Computer Science **20**(3) (2009) 427–442
10. Gheorghe, M., Ipate, F., Dragomir, C., Mierlă, L., Valencia-Cabrera, L., García-Quismondo, M., Pérez-Jiménez, M.J.: Kernel P Systems - Version 1 (2013)
11. Păun, G., Rozenberg, G., Salomaa, A., eds.: The Oxford Handbook of Membrane Computing. Oxford University Press (2009)
12. Frisco, P., Gheorghe, M., Pérez-Jiménez, M.J., eds.: Applications of Membrane Computing in Systems and Synthetic Biology. Springer (2014)
13. Romero-Campero, F.J., Twycross, J., Cao, H., Blakes, J., Krasnogor, N.: A multiscale modeling framework based on P systems. In: Membrane Computing. Volume 5391 of LNCS. Springer (2009) 63–77
14. Gillespie, D.: A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. Journal of Computational Physics **22**(4) (1976) 403–434
15. Blakes, J., Twycross, J., Romero-Campero, F.J., Krasnogor, N.: The Infobiotics Workbench: An integrated in silico modelling platform for systems and synthetic biology. Bioinformatics **27**(123) (2011) 3323 – 3324
16. Blakes, J., Twycross, J., Konur, S., Romero-Campero, F.J., Krasnogor, N., Gheorghe, M.: Infobiotics Workbench: A P systems based tool for systems and synthetic biology. In: [12]. Springer (2014) 1–41
17. Dragomir, C., Ipate, F., Konur, S., Lefticaru, R., Mierlă, L.: Model Checking Kernel P Systems systems. In: 14th International Conference on Membrane Computing. Volume 8340 of LNCS., Springer (2013) 151–172
18. Hinton, A., Kwiatkowska, M., Norman, G., Parker, D.: PRISM: A tool for automatic verification of probabilistic systems. In: Proc. TACAS. Volume 3920 of LNCS. Springer (2006) 441–444
19. Donaldson, R., Gilbert, D.: A Monte Carlo model checker for probabilistic LTL with numerical constraints. Technical report, Bioinformatics Research Centre, University of Glasgow, Glasgow (2008)
20. Holzmann, G.J.: The model checker SPIN. IEEE Transactions on Software Engineering **23**(5) (1997) 275–295

21. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects of Computing **6** (1994) 102–111
22. Baier, C., Haverkort, B., Hermanns, H., Katoen, J.: Model-checking algorithms for continuous-time Markov chains. IEEE Transactions on Software Engineering **29**(6) (2003) 524–541
23. Terrazas, G., Lui, L.T., Krasnogor, N.: Spatial computation and algorithmic information content in non-DNA based molecular self-assembly. In: 6th International Workshop on Spatial Computing. (2013) 85–90