ACS SyntheticBiology

Tutorial

pubs.acs.org/synthbio

Qualitative and Quantitative Analysis of Systems and Synthetic Biology Constructs using P Systems

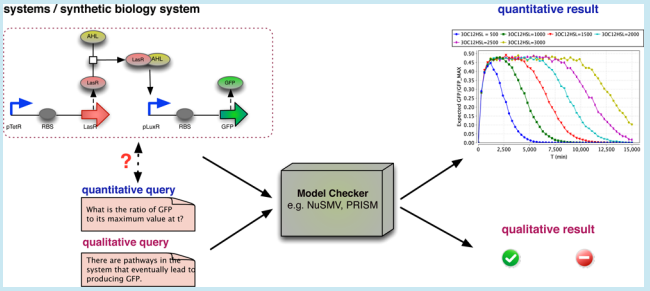
Savas Konur,*[,†] Marian Gheorghe,[†] Ciprian Dragomir,[†] Laurentiu Mierla,[‡] Florentin Ipate,[‡] and Natalio Krasnogor[§]

[†]Department of Computer Science, University of Sheffield, Sheffield, S1 4DP, United Kingdom

[‡]Department of Computer Science, University of Bucharest, Bucharest, 060042, Romania

[§]School of Computing Science, Newcastle University, Newcastle upon Tyne, NE1 7RU, United Kingdom

ABSTRACT: Computational models are perceived as an attractive alternative to mathematical models (e.g., ordinary differential equations). These models incorporate a set of methods for specifying, modeling, testing, and simulating biological systems. In addition, they can be analyzed using *algorithmic* techniques (e.g., *formal verification*). This paper shows how formal verification is utilized in systems and synthetic biology through *qualitative* vs *quantitative* analysis. Here, we choose two well-known case studies: quorum sensing in *P. aeruginosas* and pulse generator. The paper reports verification analysis of two systems carried out using some model checking tools, integrated to the Infobiotics Workbench platform, where system models are based on stochastic P systems.



Advances in our understanding of chemical and biological complexity are posed to open up a new frontier in computing science. Indeed, what we like to call "algorithmic living matter" will lead to a new Bio-ICT revolution based on the ability to program all kinds of materials in all kinds of environments at all scales, with engineered biological cells as the primary workhorse of this brewing revolution. We are still, however, far away from the holy grail of algorithmic living matter. This is so because, notwithstanding tremendous progress in synthetic biology, we still lack the tools, both biotechnological and computational, that are required to deliver highly calibrated and standardized bioparts and devices that could be pluged-and-played seamlessly, scalably, and dependably.

In software engineering, the concept of dependability[1,2] includes such properties as reliability, robustness, and safety of software systems. In this paper, we are concerned with dependability and, in particular, dependability of biomodels.

Model checking[3] is at the cornerstone of software dependability. Model checking is a computational technique exhaustively checking that a system satisfies certain requirements. All extant tools for model checking require as a starting point a formal definition of "the system" that will be analyzed and queried. The method has been extensively used in the verification of various systems. Recently, it has been also applied to formal analysis of biomodels (e.g., ERK/MAPK pathway[4]). In order to initiate a wider exploration of these concepts within the synthetic biology community, we focus on state-of-art model checking tools and explore their potential and limitations for synthetic biology.

We have shown in ref 5 that *stochastic P systems*[6] are an amenable formalism to capture a biomodel, as it is both intuitive (as it follows a rule-base modeling philosophy) and well structured. The later is important as it allows us to automatically translate a biomodel specification written in stochastic P systems to a model checking domain specific language (DSL) in a rather straightforward manner.

In this paper, we take that approach; namely, we specify two well-known biological systems, one arising from systems biology (*P. aeruginosas* quorum sensing) and one from synthetic biology (pulse generator). We use the Infobiotics Workbench tool,[7] which is based on stochastic P systems, to model the systems and formally analyze them using some model checkers integrated into the workbench. We show how formal verification is utilized in systems and synthetic biology by interplaying qualitative vs quantitative analysis. We also address scalability issues using two case studies.

The paper is organized as follows: We will first provide an overview of Infobiotics Workbench and stochastic P systems. This is followed by a brief description of model checking. After describing the quorum sensing and pulse generator systems, we will discuss the model checking experiments carried out and analyze the results. We will finally draw conclusions and suggest areas of future work.

INFOBIOTICS WORKBENCH

The Infobiotics Workbench (Ibw) tool[8] permits rapid biomodel prototyping and specification, simulation, verification,

Special Issue: SEED 2014

Received: March 7, 2014
Published: August 4, 2014

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92

analysis, and optimization. This is an integrated software suite consisting of individual components responsible for various functions. These are executed from the *Infobiotics Dashboard*, which also helps visualizing results in time and space.[7]

In Ibw, system models are constructed using a rule-based specification language, called *stochastic P systems*. Stochastic P systems are utilized to specify (multi)cellular systems and molecular interactions taking place in different locations of the living cells or across cells within a population. With these models there is a clear mapping of different locations of biological systems into "compartments" delimited by membranes that have a formal representation within the stochastic P system language. Each molecular species is associated with an object in the multiset corresponding to a membrane mapping the region or compartment where the molecule is located. Depending on the complexity of these species, various object types are utilized, including simple symbols or strings. Strings are also utilized to represent the genetic information encoded by macromolecules such as RNA, DNA, proteins, etc. Different simple molecular interactions or more complex gene expression, compartment translocation, as well as cell division and death are specified using object and string rewriting or communication, and compartment division or dissolution. These rules deal with multisets of objects rather than objects.

Many P systems use these features in building a model. In the case of a stochastic P systems, constants, called *kinetic constants*, are associated with rules in order to compute their probabilities and time needed to be applied, according to Gillespie algorithm.[9] 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. Stochastic P systems representing models of individual living cells can be associated with different geometries or topologies for modeling multicellular systems with their interactions. One such geometry is a grid of cells that can be represented by a *lattice*, and inside each component of it resides a membrane with stochastic behavior. In this case, the communication is only among neighbors. In our examples, we will use stochastic P systems containing only simple objects, unicellular or multicellular structures. In the pulse generator system, we also use a lattice to represent the geometry.

Related Work. We have shown[10] that stochastic P systems can be linked not only to stochastic algorithms but also to coarsed grained molecular simulations (e.g., Dissipative Particle Dynamics) in order to model chemical processes taking place inside chemical miscelles and vesicles. In ref 5, we have shown how this biolanguage can be used to specify increasingly complex models by reuse of rules' "modules" capturing key biological processes such as transcription regulation, enhanced degradation rates due to addition of degradation tag fusions to gene products, riboswitch controls, etc. Furthermore, due to its well structured syntax and clear semantics, it is possible to apply *in silico* directed evolution over the specifications in stochastic P systems and evolve circuits matching specific phenotypes.[11,12]

MODEL CHECKING

Model checking is an algorithmic technique that formally demonstrates the correctness of a system by means of strategic and exhaustive state space investigation. The technique is based on building a mathematical model of the system in question. The properties to be verified are expressed in specific logical expressions, called *temporal logic formulas*, and then checked against the model.

Model checking tools allow analyzing various temporal logic properties in an automatic way. The method has been extensively used in the verification of various systems, for example, concurrent[13] and distributed systems,[14] multiagent systems,[15] pervasive systems,[16] and swarm robotics.[17] Recently, it has been also applied to the analysis of various biological systems, such as ERK/MAPK pathway,[4] FGF signaling pathway,[18] cell cycle in eukaryotes,[19] EGFR pathway,[20] T-cell receptor signaling pathway,[21] cell cycle control,[22] diabetes-cancer signaling network,[23] and genetic Boolean gates.[24,25]

Two types of verification approaches are utilized in this paper, a qualitative analysis and a quantitative one.[4] In the former, all the possible behavior pathways are considered and the presence or absence of different species, in certain conditions, is detected or certain sequences of events are verified. In the latter, the appearance of certain amounts of species is checked in accordance with certain conditions.

Figure 1 shows the taxonomy of the model checkers used in this paper.

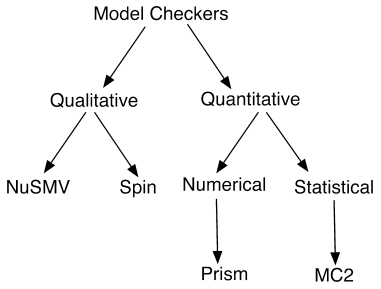


Figure 1. Taxonomy of the model checking tools discussed in the paper.

Qualitative Model Checking. A model checker requires an unambiguous representation of its input model, together with a set of correctness claims (i.e., properties) generally expressed as temporal logic formulas. In qualitative model checking, the correctness result is returned categorically (i.e., a "yes' or 'no" answer). In case of "no", a counter-example is returned to help users find out why the model does not satisfy the correctness claim.

In this paper, we explore the use of two qualitative model checkers: Spin and NuSMV. Spin[26] is a widely used model checking tool and is particularly suited for models of concurrent and distributed systems, described by means of interleaved atomic instructions. It features a high level modeling language, called Promela, which specializes in concise descriptions of concurrent processes and interprocess communication. A practical and convenient aspect of the language is the use of discrete primitive data types and custom data types similar to the those in C, allowing fine grain model details or low level implementation features to be directly expressed as part of the model. Spin provides complete support for *Linear-time Temporal Logic* (LTL) and its *on the fly* verification procedure which avoids the necessity to generate the global state space prior to the search for satisfiability.

NuSMV[27] is another popular model checking tool, designed to verify synchronous and asynchronous systems. NuSMV's high-level modeling language is based on *Finite State Machines* (FSM) and allows the description of systems in a modular and hierarchical manner. NuSMV supports the analysis of specification expressed in *Linear-time Temporal Logic* (LTL)

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92

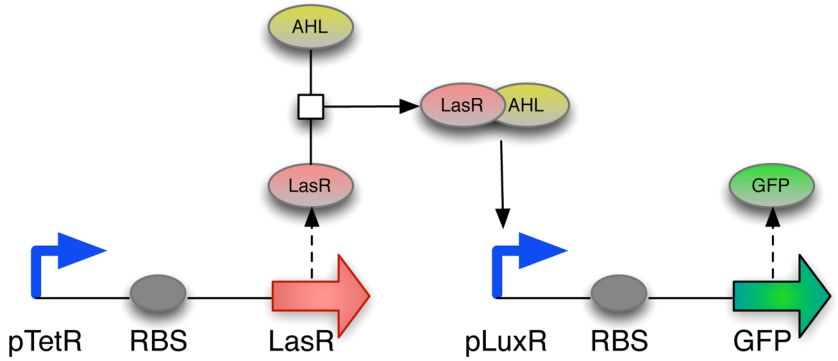


Figure 2. Genetic parts and design of the quorum sensing model (The figure is reproduced with permission from ref 34. Copyright 2013, Elsevier.).

and *Computation Tree Logic* (CTL). NuSMV employs *symbolic* methods, allowing a compact representation of the state space to increase the efficiency and performance. The tool also permits conducting simulation experiments over the provided FSM model by generating traces either interactively or randomly.

Quantitative Model Checking. As described above, qualitative model checkers can only provide categorical information, that is, a "yes' or 'no" answer. However, for most systems, especially synthetic biology models, we want to know more about the quantitative nature of the system—concentration of certain molecular species, temporal and spatial relationships involving such values, dependencies between them.

Quantitative analysis provides us more precise and valuable information about the system dynamics. *Probabilistic model checking* augments classical model checking with numerical analysis techniques to infer quantitative information about system properties, such as "the likelihood that a protein will be synthesized". The numerical characteristic of quantitative properties allows us to observe any trends or abnormalities in the system behavior.

In this paper, we will use Prism and MC2 as quantitative model checkers. Prism[28] is the mostly widely used probabilistic model checker. The tool supports several models, such as discrete time Markov chains (DTMCs), Markov decision processes (MDPs), and continuous time Markov chains (CTMCs). The tool accepts system models in a dedicated state-based high-level language. The Prism language is mainly composed of a set of state (and global) *variables*, *commands*, and *modules*. Each module represents a part of the system. Modules interact with each other and they collectively define the overall system behavior.

As for probabilistic properties, Prism supports PCTL[29] (a probabilistic extension of CTL) and Continuous Stochastic Logic (CSL[30]). Both languages employ special operators to express quantitative information. As well as numerical methods, Prism employs symbolic methods to improve the model checking efficiency.

Despite its usefulness in system analysis, model checking has a major drawback: *scalability*. Since model checking is essentially an exhaustive state space search, it cannot cope with very large state spaces. In another words, the model size has a big impact on the feasibility of the approach.

To overcome this issue, an alternative model checking approach, *statistical model checking*, has been proposed.[31] Unlike standard model checking approach, statistical model checking does not require numerical and symbolic techniques. The idea is that a number of simulation traces are generated, and the correctness is approximated using statistical methods (e.g., Monte Carlo). This implies that we cannot obtain the exact verification results, but the accuracy can only be calculated to a specified degree of *confidence*, based on the number of independent simulation runs. Although the accuracy is not as precise as the numerical approach, the performance of the model checking process is immensely improved.

As well as (numerical and symbolic) probabilistic model checking, Prism also supports statistical model checking using its *discrete event simulator*. However, the statistical version of the tool allows only a restricted set of property types to be model checked. For example, we cannot model check *steady-state* properties.

MC2[32] is another statistical model checker that allows a richer set of properties to be model checked. As well as the full formula set of probabilistic temporal logic, we can also express properties regarding the quantities such as maximum/minimum values of a species and decrease/increase of concentrations. To carry out *statistical* model checking using MC2, we can reuse previous simulation results, or can generate a larger set of simulation traces to achieve higher confidence in the model checking results.

Both Prism and MC2 are integrated to the Ibw platform, which constructs the necessary model checking inputs automatically. However, NuSMV and Spin are not integrated at the moment. In addition to a formal representation of models, model checking tools also require system properties to be verified in a logical formal syntax. Ibw also features a *natural language query* tool with a GUI which allows formal properties expressed in natural language using some predefined patterns.[7]

BIOLOGICAL MODELS

In this section, we will describe two biological models to be analyzed using formal verification. We have chosen two well-known cases, one derived from systems biology (quorum sensing in *P. aeruginosas*) and one derived from synthetic biology (pulse generator). We do this so we can focus on our analysis methods rather than in the intricacies of a putative new biological system.

Quorum Sensing. Our first model is a well-known example in systems biology, *quorum sensing*. Quorum sensing (QS) is a communication mechanism of bacteria, working based on some signaling molecules. Once the QS process is activated, the concentration of the signaling molecule is an indicator of the number of cells in the colony.[33] QS has been designed and studied in numerous papers. Here, we will focus on the construct designed by Saeidi et al.[34] The authors designed a

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92

quorum sensing device that can sense the pathogenic *Pseudomonas aeruginosa* bacteria, and upon detection, it illuminates green fluorescent protein (GFP).

Figure 2 illustrates the quorum sensing device and its corresponding design. The system can be briefly described as follows:[34]

"The TetR promoter (pTetR), which is constitutively on, produces a transcriptional factor (LasR) that binds to AHL (3OC12HSL). The LuxR promoter (pLuxR), to which LasR-AHL activator complex binds, serves as the inducible promoter in our sensing system. The formation of the LasR-AHL complex, which binds to the LuxR promoter, leads to the production of GFP as the reporter protein."

To analyze different aspects of the quorum sensing system, we consider two variations of the model: a rule-based *stochastic* model and *nondeterministic* model.

Stochastic Model. The stochastic model comprises a set of rules that govern the kinetic and stochastic behavior of the system. The reaction rules and the corresponding kinetic rate contents are provided in Table 1.

Table 1. Reaction Rules for the QS Model

rule		stochastic constant
r_{1a}	Ptet_LasR $\xrightarrow{k_{1a}}$ mRNA_LasR	$k_{1a} = 3.734\ \text{min}^{-1}$
r_{1b}	mRNA_LasR $\xrightarrow{k_{1b}}$ Ptet_LasR	$k_{1b} = 3.48 \times 10^{-1}\ \text{min}^{-1}$
r_2	mRNA_LasR $\xrightarrow{k_2}$ LasR	$k_2 = 35.7\ \text{min}^{-1}$
r_3	LasR $\xrightarrow{k_3}$	$k_3 = 6.96 \times 10^{-2}\ \text{min}^{-1}$
r_{4a}	LasR + AHL $\xrightarrow{k_{4a}}$ LasR-AHL	$k_{4a} = 9.6 \times 10^{6}\ \text{min}^{-1}$
r_{4b}	LasR-AHL $\xrightarrow{k_{4b}}$ LasR + AHL	$k_{4b} = 0.0\ \text{min}^{-1}$
r_{5a}	LasR-AHL + pLuxR $\xrightarrow{k_{5a}}$ A_pluxRR	$k_{5a} = 1.96 \times 10^{6}\ \text{min}^{-1}$
r_{5b}	A_pluxRR $\xrightarrow{k_{5b}}$ LasR-AHL + pLuxR	$k_{5b} = 10.2\ \text{min}^{-1}$
r_6	AHL $\xrightarrow{k_6}$	$k_6 = 2.832 \times 10^{-4}\ \text{min}^{-1}$
r_7	A_pluxRR $\xrightarrow{k}$ GFP	k

In Table 2, the kinetic constant k is dynamically calculated at each step according to the following formula:[34]

Table 2. Kinetic Constants for the Rule r_7

constant	value
k_7	4.051×10^{-3}
k_8	9.567×10^{-3}
k_9	9.742×10^{-8}
k_{10}	6.5×10^{-16}
k_{11}	$1.0 \times 10^{-14}\ \text{min}^{-1}$
k_{12}	$2.4 \times 10^{-7}\ \text{min}^{-1}$
n_1	2.0
n_2	2.0

$$k = \left(k_7 + \frac{k_8[\text{A\_pluxRR}]^{n_1}}{k_9^{n1} + [\text{A\_pluxRR}]^{n_1}}\right)\left(k_{10} + \frac{k_{11}[\text{A\_pluxRR}]^{n_2}}{k_{12}^{n2} + [\text{A\_pluxRR}]^{n_2}} - [\text{GFP}]\right)$$

where the entity $[a]$ denotes the concentration of the species a. The reaction constants used in the formula above is given in Table 2.

Nondeterministic Model. The nondeterministic model is directly derived from the stochastic model by taking out the kinetic constants. This model describes all the interactions provided by its stochastic counterpart, but in a rather symbolic and qualitative way than showing more precise quantitative aspects of the system. All the possible pathways of the system are captured by the model, but as the precise concentration of certain molecular species is not a key issue for these models, in certain circumstances the multisets are bounded, even restricted to one or two elements, describing their presence rather than their molecular concentration.

Pulse-Generator Model. The *pulse generator* is a colony of synthetically programmed bacteria, studied in Basu et al.[35] The system is composed of two sets of *sender* and *pulsing* cells (see Figure 3). The sender cells produce a signaling molecule, propagated through the pulsing cells. The pulsing cells express the green fluorescent protein (GFP) triggered by the signaling molecules, and propagate the excess signaling molecules to the neighboring cells.

Sender cells synthesize the signalling molecule 3OC6-HSL (AHL) through the enzyme LuxI, expressed under the constitutive expression of the promoter PLtetO1. Pulsing cells express GFP under the regulation of the PluxPR promoter, activated by the LuxR_3OC6_2 complex. The LuxR protein is expressed under the control of the PluxL promoter. The GFP production is repressed by the transcription factor CI, codified under the regulation of the promoter PluxR that is activated upon binding of the transcription factor LuxR_3OC6_2.

In a colony, sender bacteria strains are located at one end of a specific spatial distribution (e.g., lattice), and pulsing cells are located to the rest of the distribution (lattice).

Stochastic Model. The stochastic behavior of the pulse generator system is modeled using a set of stochastic P system models. The language permits modelling a two-dimensional geometric lattice where "a population of stochastic P systems could be placed and over which molecules could be *translocated*".[7]

Each sender cell contains 11 reaction rules, which model the production of the signaling molecule 3OC6-HSL. Each pulsing cells contains 38 reaction rules, which models the production of the pulse of GFP protein as a response to the signal 3OC6-HSL. The geometry of a bacterial colony containing both cell types is represented by a rectangular lattice. The overall model of the pulse generator system is captured by "distributing cellular clones of the sender cell strain at one end of the lattice and cellular clones of the pulsing cell strain over the rest of the points".[7]

Nondeterministic Model. As for the previous model, the kinetic constants are removed. In the case of a stochastic P systems (with spacial information), the individual component models are compacted such that an entire chain of reactions is replaced by one or a few simple rules. Consequently, the overall number of interactions is reduced, and all the species that do not appear in the new set of rules are removed from the model. These changes are made in the nondeterministic models as these are used for qualitative analyses where the concentration of certain molecules is not significant or chain of reaction already analyzed can be replaced by some abstractions mimicking their behavior through simpler rewriting mechanisms.

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92

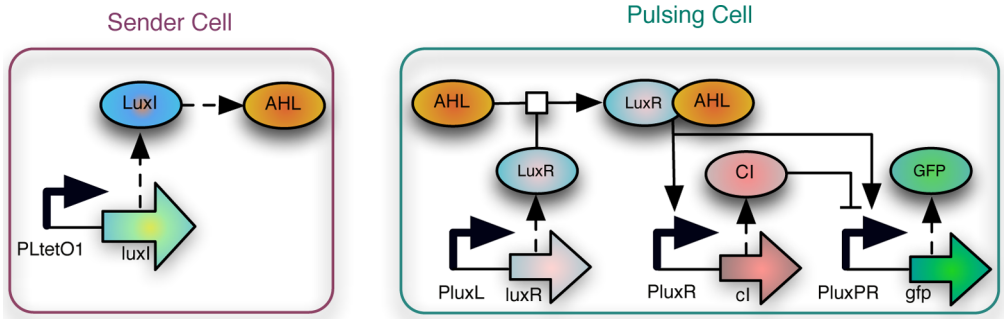


Figure 3. Sender and pulsing cells of the pulse generator system (The figure is reproduced from ref 7. Copyright 2014, Springer.).

EXPERIMENTS

In this section, we will discuss formal qualitative analysis of one systems and one synthetic biology construct using the model checkers NuSMV (for the quorum sensing model) and Spin (for the pulse generator model) to capture qualitative aspects of the system, and quantitative analysis using the probabilistic model checkers Prism and MC2 to capture stochastic and kinetic dynamics.

Before discussing the experiments carried out, we present in Table 3 the main symbols used in temporal logic formulas.

Table 3. Main Symbols in Temporal Logic Formulas

temporal operator	informal meaning
F p	p holds in some future state
G p	p always holds in all future states
p U q	p holds until q holds
Brach quantifier	informal meaning
E	p for some paths (i.e., executions)
A	p for all paths (i.e., executions)

Generally speaking, a property is constructed as follows: branch quantifier + temporal operator + expression (e.g., AG (GFP $\geq$ *thr*)).

In probabilistic temporal logics, we represent probabilistic properties with formulas of the type $P_{\bowtie r}[\varphi]$, which informally means that the probability of taking a branch satisfying φ meets $\bowtie r$ (where $\bowtie \in \{<, >, =, \leq, \geq\}$ and $r \in [0,1]$). In addition to the probabilistic formulas, the Prism model checker also permits expressing *reward*-based formulas to calculate expected values. For example, the reward formula $R\{\text{“X”}\}_{=?}[I = t]$ calculates the expected value of X at the time instant t.

We note that the complete model and experimental results of the pulse generator can be downloaded from ref 36. These include LPP model files, simulation parameters, simulation results, Prism model file, model checking parameters, a list of Prism properties and model checking experimental results. The experimental results and source files of the quorum sensing model can also be accessed at ref 37.

Quorum Sensing Model. The formal analysis of the quorum sensing system concerns a single sensing device. The qualitative model checking experiments refer to relationships between species occurring on various reaction pathways, and the quantitative model checking experiments refer to more complex stochastic behaviour of the system.

Qualitative Model Checking. In accordance with the genetic parts and the design of the quorum sensing system presented in Figure 2, the model consists of a chain of reactions. For instance, one can verify the appearance of GFP later in the system, but not at the beginning, or the fact that the protein LasR binds to the signal molecule AHL (3OC12HSL) and the obtained complex, which binds to the LuxR promoter, will eventually lead to the production of the GFP reporter protein. These are expressed as follows by using formulas expressed in LTL or CTL for the model checker chosen in this respect.

Property 1. There are pathways in the system that eventually lead to producing GFP.

$$\text{EF}\ (\text{GFP} > 0)$$

This property, expressed in CTL and verified in NuSMV, is true, as expected.

We now show how one can check that a certain sequence of events must or might appear in a chain of reactions. This is illustrated by the relationship between the formation of the complex LasR_AHL and the production of the GFP, by using a CTL formula in NuSMV.

Property 2. Always the Lasr_AHL complex formation might eventually lead to the production of GFP.

$$\text{AG}\ (\text{LasR\_AHL} > 0 \Rightarrow \text{EF GFP} > 0)$$

The result of this property is true and shows that the production of GFP is always preceded by the Lasr_AHL complex formation.

Quantitative Model Checking. Here we report the results of the quantitative analysis of the quorum sensing model. The experiments have been carried out using Prism and MC2.

Prism Results. We first start with the Prism results. Note that for computational reasons (as discussed), we have performed our experiments using the statistical (approximate) model checking module of Prism, based on the discrete event simulator of the tool.

In the following, we give both informal and formal representations of the properties checked:

Property 1. What is the probability that the system produces GFP if there is not any 3OC12HSL?

The formal translation of this property in the Prism's property language CSL is given as follows:

$$P_{=?}[3\text{OC12HSL} = 0\ \text{U GFP} > 0]$$

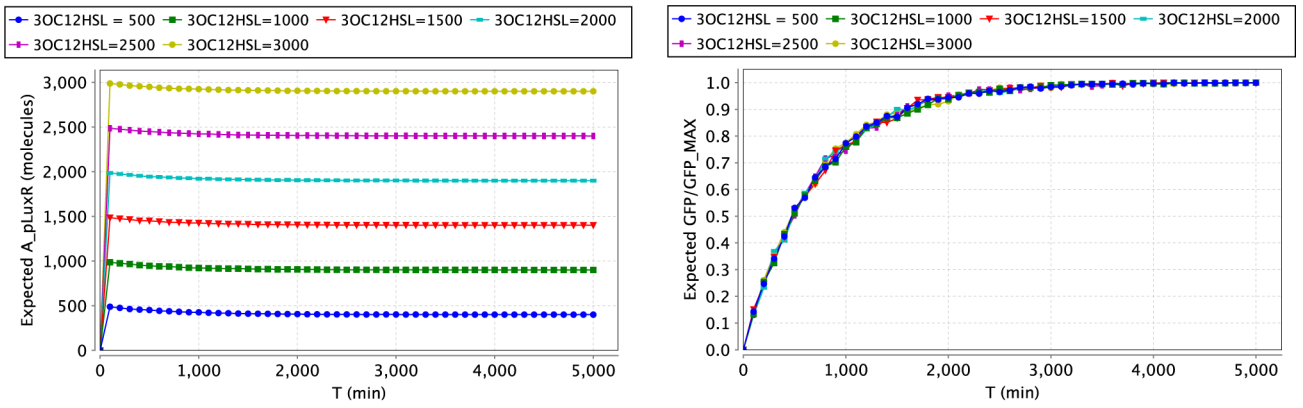
The result of this property is 0.0, which ensures that the system will not produce GFP in the absence of *Pseudomonas*.

Property 2. What is the probability that the production of GFP starts in t minutes (if 3OC12HSL is available initially)?

The property is formally translated as follows:

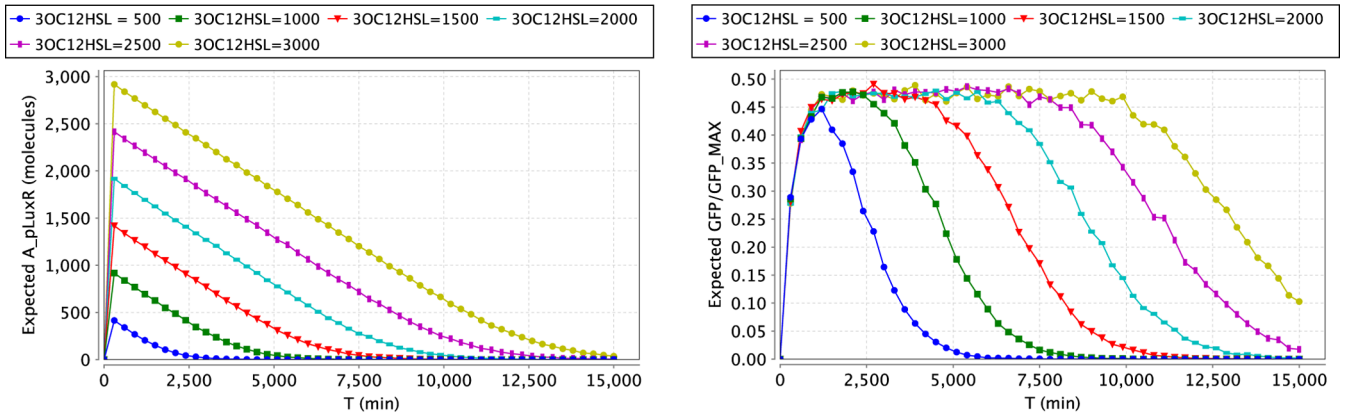
$$P_{=?}[F^{\leq t}\ \text{GFP} > 0]$$

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92



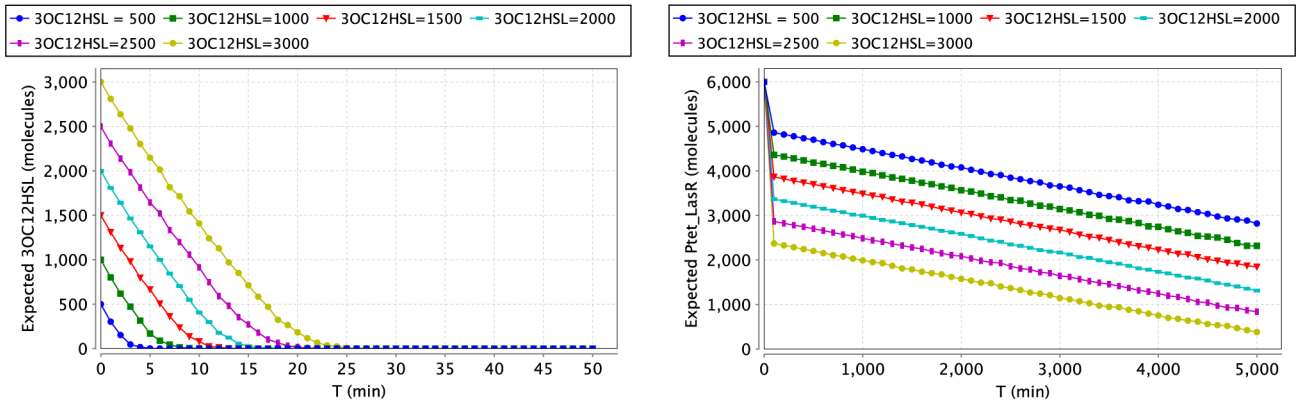
(a) A_pluxR vs 3OC12HSL (b) GFP/GFP_ratio vs 3OC12HSL

Figure 4. Expected amount of species in the quorum sensing model based on different initial amounts of 3OC12HSL.



(a) A_pluxR vs 3OC12HSL (b) GFP/GFP_MAX vs 3OC12HSL

Figure 5. Expected amount of species in the modified model based on different initial amounts of 3OC12HSL.



(a) 3OC12HSL (b) Ptet_LasR vs 3OC12HSL

Figure 6. Expected amount of 3OC12HSL and Ptet_LasR for both models.

The results obtained for $t = 100$ and $t = 1000$ are 0.13 and 0.76, respectively. As expected, it takes some time the system to fuse GFP after sensing the bacteria.

Property 3. What is the expected A_PluxR at the time instant t?

The property is formally expressed in CSL:

$$R\{\text{“A\_PluxR”}\}_{=?}[I = t]$$

The results based on different initial values of 3OC12HSL are illustrated in Figure 4a. The figure shows that the concentration of the active complex A_PluxR, producing GFP, depends on the signaling molecule 3OC12HSL. The results are inline with the original design of the device.[34]

Property 4. What is the ratio of GFP to its maximum value at the time instant t?

The property is translated as

$$R\{\text{“GFP/GFP\_MAX”}\}_{=?}[I = t]$$

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92

The results are presented in Figure 4a, where GFP_MAX denotes the maximum concentration that GFP can reach. The results show that no matter the initial concentration of 3OC12HSL, GFP can reach its maximum level at around 3000 min.

We can conclude from Figure 4 that the QS device emits GFP in its highest level, even if the 3OC12HSL's concentration reduces to 0 in 25 min (see Figure 6a). Naturally, a sensing device should stop fusing GFP when there is not any signaling molecule left (i.e., no bacteria are available). To implement this effect, we have modified the QS device slightly by adding two degradation rules for GFP and A_PluxR. Based on these changes, we have reanalyzed the properties 3 and 4. As shown in Figure 5, the concentration levels of the both molecules starts decreasing after all 3OC12HSL molecules are consumed.

Similar to Properties 3 and 4, we have also analyzed the expected number of 3OC12HSL and Ptet_LasR. As expected, for both original and modified models, we have obtained the same results (see Figure 6).

MC2 Results. As discussed previously, the approximate model checking module of Prism allows only a restricted set of property types to be model checked. MC2 permits expressing a richer set of properties. As well as the full formula set of probabilistic temporal logic, we can also express properties regarding quantitative aspects such as maximum/minimum values of a species and decrease/increase of concentrations.

Below, we provide some properties analyzed (against the original system) using MC2 and present the corresponding model checking results. Note that Prism's statistical model checker does not support these properties. Here, we assume that the signaling molecule is initially available.

Property 1. Probability that the GFP production is activated whenever the signaling molecule becomes available.

The formal translation of Property 1:

$$P_{=?}[G\ ([3OC12HSL] > 0 \Rightarrow F\ [GFP] > 0)]$$

The result of this property is 1.0, which ensures that the system will produce GFP whenever *Pseudomonas* is sensed.

Property 2. Probability that the GFP concentration eventually increases.

The formal translation of Property 2:

$$P_{=?}[F\ (d[GFP] > 0)]$$

The result of this property is 1.0, which ensures that the system responds to 3OC12HSL.

Property 3. Probability that the GFP concentration increases within 1000 min.

The formal translation of Property 3:

$$P_{=?}[F\ (d[GFP] > 0 \wedge time \leq 1000)]$$

The result of this property is 0.84, implying it is very likely that the system will produce GFP within 1000 min of sensing 3OC12HSL.

Property 4. Probability that the maximum GFP concentration is reached within 1000 min.

The formal translation of Property 4:

$$P_{=?}[F\ [GFP] = \max[GFP] \wedge time \leq 1000]$$

The result of this property is 0.80, implying it is likely that the system will respond to 3OC12HSL very strongly within 1000 min.

Property 5. Probability that GFP is fused in the steady-state.

The formal translation of Property 5:

$$P_{=?}[F\ G\ [GFP] > 0]$$

The result of this property is 1.0.

We can obtain similar results for the modified model except Property 5, for which we have obtained 0.0. These results show that in the original model the GFP production is activated as a response to the availability of the signaling molecule and it never stops. However, in the modified model, the GFP production is stopped after the signaling molecules are totally consumed.

Pulse-Generator Model. This is a model of a multicellular system showing how the verification method discussed so far for a cell system can be extended to more complex cases, by avoiding the well-known state explosion problem that all the model checking approaches face. First, for the qualitative analysis we are focusing on two aspects: the verification of individual components of the system and the verification of the entire system, or a significant fragment of it, by using adequate modeling abstractions. Second, the quantitative approach will show how statistical model checking can be used to verify properties referring to molecules from different parts of the system.

Qualitative Model Checking. We start with verifying properties similar to those presented in the section on qualitative model checking for the quorum sensing system.

Property 1. There is no pathway in the sending cell that eventually leads to the production of signal3OC6 in less than k steps.

$$G\ (step < k \Rightarrow signal3OC6 = 0)$$

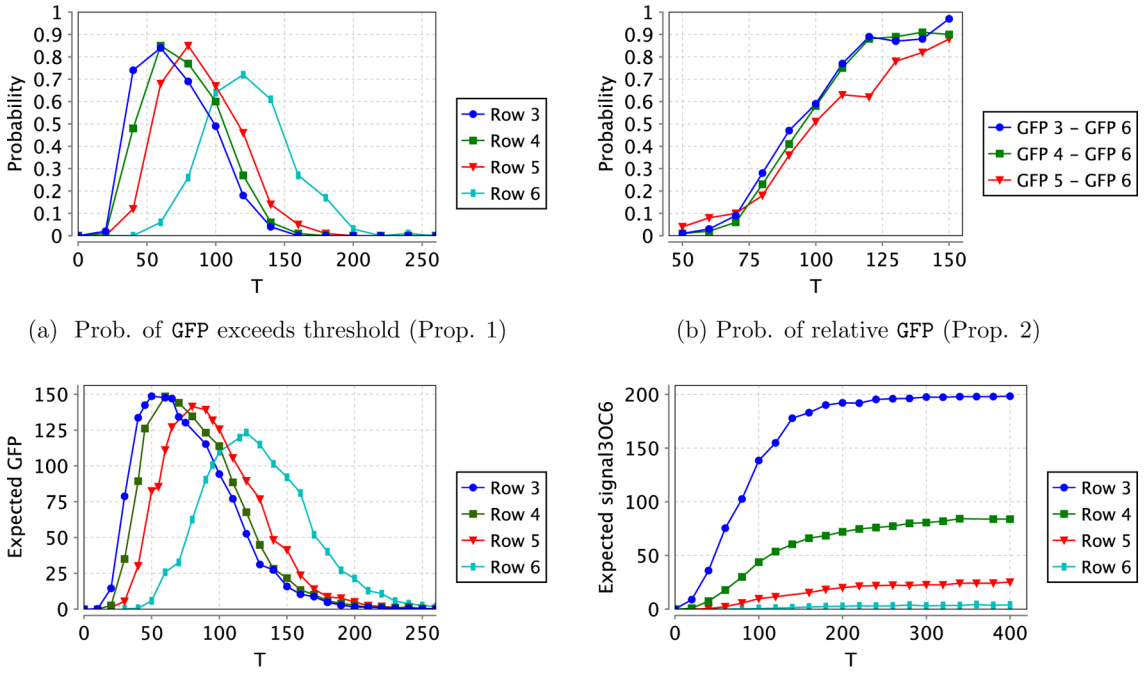
The property, expressed in LTL, has been verified in Spin, which returned true for $k = 5$. This type of properties are useful to acquire information about the reaction network of a particular system. We do not repeat the experiments here, but one can show, similar to the quorum sensing case, the presence of the molecules in different sequences of events.

For the multicellular case of the lattice system, we check whether the signal molecule properly propagates through the system. Running such experiments, even for simple lattices of 2 by 2, is very prohibitive in both time and memory requested. This shows that one needs a different representation for the model used. One solution to overcome this issue is to compact some long chains of reactions into simpler and shorter ones where only the molecules directly involved in verification are kept. As we aim to check the propagation of the signal molecules through the lattice, we use reduced models for the two cell types involved. The sender cell has now 5 rules and the pulsing cell has 7 rules. As the sending cells produce the AHL (signal3OC6) signal molecule, the propagation of the AHL molecule is verified only for the pulsing cells. For the experiments in this section the lattice will be reduced to a 2 by 4 size with two sender cells in the first row and pulsing cells in the rest.

Property 2. The presence of the GFP at row $n \in \{2,3,4\}$ is preceded by the presence of signal3OC6 in the preceding row, $n' = n - 1$, or neighbor cell.

The property is translated as

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92



(a) Prob. of GFP exceeds threshold (Prop. 1)

(b) Prob. of relative GFP (Prop. 2)

(c) Expected GFP protein (Prop. 3)

(d) Expected signal3OC6 (Prop. 4)

Figure 7. Quantitative analysis using probabilistic model checking (Taken from[7]. Copyright 2014, Springer.). Row n denotes the nth row of the pulsing cells in the lattice, and T denotes time.

$$(\mathrm{F}\ \mathrm{GFP\_pulsing\_n} > 0) \Rightarrow \neg\ ((\mathrm{signal3OC6\_n}' = 0 \wedge \mathrm{signal3OC6\_n} = 0)\mathrm{U}\ \mathrm{GFP\_pulsing\_n} > 0)$$

We have verified this property, expressed in LTL, using Spin, which returned true.

Quantitative Model Checking. We now present some numerical results based on quantitative analysis, performed using Prism. The results presented in this section are summarized from our earlier work.[7]

In the following set of experiments, we consider a lattice of size 2 by 6. The sender cells are placed to the first 2 rows and columns, that is, the initial 2 by 2 section of the lattice. The pulsing cells are distributed to the remainder of the lattice.

Property 1. What is the probability that GFP concentration at row $n \in \{3,4,5,6\}$ exceeds 100 at the time instant T?

The formal translation of this property in CSL is given as follows:

$$\mathrm{P}_{=?}[\mathrm{true}\ \mathrm{U}^{[T,T]}\ \mathrm{GFP\_pulsing\_n} \geq 100]$$

The verification results are presented as a 2D plot in Figure 7a.

Property 2. What is the probability that GFP concentration at row $n \in \{3,4,5\}$ stays greater than GFP concentration at row 6 until the time instant T where GFP concentration at row 6 exceeds GFP concentration at row n?

The property is formally translated to CSL as follows:

$$\mathrm{P}_{=?}[\mathrm{GFP\_pulsing\_n} \geq \mathrm{GFP\_pulsing\_6}\ \mathrm{U}^{[T,T]}\ \mathrm{GFP\_pulsing\_6} > \mathrm{GFP\_pulsing\_n}]$$

The verification results are illustrated in Figure 7b.

Property 3. What is the expected GFP concentration at row $n \in \{3,4,5,6\}$ at the time instant T?

The property is formally expressed as

$$\mathrm{R}\{\text{“GFP\_pulsing\_n”}\}_{=?}[\mathrm{I} = T]$$

The results are illustrated in Figure 7c.

Property 4. What is the expected signal3OC6 concentration at row $n \in \{3,4,5,6\}$ at the time instant T?

The property is translated as

$$\mathrm{R}\{\text{“signal3OC6\_pulsing\_n”}\}_{=?}[\mathrm{I} = T]$$

The corresponding verification results are shown in Figure 7d.

Based on the verification results, we can infer some important information regarding the system dynamics and kinetic behavior. Figure 7 clearly shows that the GFP expression propagates in this bacterial colony. In particular, parts a and c of Figure 7 illustrate the propagation of a wave of gene expression. It can be observed from these figures that the GFP concentration first increases in the rows closer to the sender cells, and then, it gradually reduces to zero. The rows far from the sender cells show a similar behavior with some delay. This delay is proportional to the distance of the row to the sender cells. Finally, Figure 7d shows that the pulsing cells which are distant from the sender cells are less likely to produce a pulse.

CONCLUSION AND FUTURE WORK

In this paper, we have shown how formal verification is utilized in systems and synthetic biology through qualitative vs quantitative analysis. Here, we have chosen two well-known case studies: quorum sensing in *P. aeruginosas* and the pulse generator. We have constructed the corresponding stochastic P system models using the Ibw software suit.

We have performed formal qualitative analysis of one systems and one synthetic biology construct using the model checkers NuSMV (for the quorum sensing model) and Spin (for the pulse generator model) to capture qualitative aspects of the system, and quantitative analysis using the probabilistic model

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92

checkers Prism and MC2 to capture stochastic and kinetic dynamics. Based on the verification results, we have inferred some important information regarding the system dynamics and kinetic behavior as well as the reaction network and system topology.

In our future work, we aim to broaden the spectrum of the current model checkers used in formally verifying synthetic biology systems and to identify in a more rigorous way a suitable model checker that fits the characteristics of a system and the properties that are verified in this case.

AUTHOR INFORMATION

Corresponding Author

*Email: s.konur@sheffield.ac.uk.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

S.K. and M.G. acknowledge the support provided for synthetic biology research by EPSRC ROADBLOCK project (EP/I031812/1). The work of N.K. is supported by the EPSRC ROADBLOCK project (EP/I031642/1), EPSRC AUDACIOUS project (EP/J004111/1), and FP7 STREP CADMAD project. M.G., L.M., and F.I. are partially supported by the MuVet project, (CNCS UEFISCDI), grant number PN-II-ID-PCE-2011-3-0688. C.D. was sponsored by an EPSRC studentship.

REFERENCES

(1) Kopetz, H., Laprie, J. C., Randell, B., and Littlewood, B., Eds. (1995) *Predictably Dependable Computing Systems*; Springer-Verlag, Inc., New York.

(2) Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C. (2004) Basic concepts and taxonomy of dependable and secure computing. *IEEE Trans. Dependable Secure Comput. 1*, 11−33.

(3) Baier, C. and Katoen, J.-P. (2008) *Principles of Model Checking (Representation and Mind Series)*; The MIT Press, Cambridge, MA.

(4) Heiner, M., Gilbert, D., and Donaldson, R. (2008) Petri nets for systems and synthetic biology. *Formal Methods Comput. Syst. Biol. 5016*, 215−264.

(5) Romero-Campero, F. J., Twycross, J., Camara, M., Bennett, M., Gheorghe, M., and Krasnogor, N. (2009) Modular assembly of cell systems biology models using P systems. *Int. J. Found. Computer Sci. 20*, 427−442.

(6) Romero-Campero, F. J., Twycross, J., Cao, H., Blakes, J., and Krasnogor, N. (2009) *Membrane Computing*; Lecture Notes in Computer Science; Vol. *5391*, pp 63−77, Springer, Berlin Heidelberg.

(7) Blakes, J., Twycross, J., Konur, S., Romero-Campero, F., Krasnogor, N., and Gheorghe, M. (2014) *Applications of Membrane Computing in Systems and Synthetic Biology*; Emergence, Complexity, and Computation; Vol. *7*, pp 1−41, Springer International Publishing, New York.

(8) Blakes, J., Twycross, J., Romero-Campero, F. J., and Krasnogor, N. (2011) The infobiotics workbench: An integrated in silica modelling platform for systems and synthetic biology. *Bioinformatics 27*, 3323−3324.

(9) Gillespie, D. (1976) A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comput. Phys. 22*, 403−434.

(10) Smaldon, J., Romero-Campero, F., Fernandez Trillo, F., Gheorghe, M., Alexander, C., and Krasnogor, N. (2010) A computational study of liposome logic: Towards cellular computing from the bottom up. *Syst. Synth. Biol. 4*, 157−179.

(11) Romero-Campero, F. J., Cao, H., Camara, M., and Krasnogor, N. (2008) Structure and parameter estimation for cell systems biology models. *Proc. 10th Annu. Conf. Genet. Evol. Comput.*, 331−338.

(12) Cao, H., Romero-Campero, F. J., Heeb, S., Camara, M., and Krasnogor, N. (2010) Evolving cell models for systems and synthetic biology. *Syst. Synth. Biol. 4*, 55−84.

(13) Alur, R., McMillan, K., and Peled, D. (2000) Model-checking of correctness conditions for concurrent objects. *Inform. Comput. 160*, 167−188.

(14) Yabandeh, M. (2011) *Model checking of distributed algorithm implementations*. Ph.D. thesis, EFPL IC, Ecublens, Switzerland.

(15) Konur, S., Fisher, M., and Schewe, S. (2013) Combined model checking for temporal, probabilistic and real-time logics. *Theoretical Computer Science 503*, 61−88.

(16) Konur, S., Fisher, M., Dobson, S., and Knox, S. (2014) Formal verification of a pervasive messaging system. *Formal Aspects of Computing 26*, 677−694.

(17) Konur, S., Dixon, C., and Fisher, M. (2012) Analysing robot swarm behaviour via probabilistic model checking. *Robotics Autonomous Syst. 60*, 199−213.

(18) Heath, J., Kwiatkowska, M., Norman, G., Parker, D., and Tymchyshyn, O. (2008) Probabilistic model checking of complex biological pathways. *Theor. Comput. Sci. 319*, 239−257.

(19) Romero-Campero, F. J., Gheorghe, M., Bianco, L., Pescini, D., Perez-Jimenez, M. J., and Ceterchi, R. (2006) *Membrane Computing*; Lecture Notes in Computer Science; Vol. *4361*, pp 477−495, Springer, Berlin Heidelberg.

(20) Eker, S., Knapp, M., Laderoute, K., Lincoln, P., Meseguer, J., and Sonmez, K. (2002) Pathway logic: Symbolic analysis of biological signaling. *Proc. Pacific Symp. Biocomput.*, 400−412.

(21) Clarke, E. M., Faeder, J. R., Langmead, C. J., Harris, L. A., Jha, S. K., and Legay, A. (2008) Statistical Model checking in BioLab: Applications to the automated analysis of T-cell receptor signaling pathway. *Proc. 6th Int. Conf. Comput. Methods Syst. Biol.*, 231−250.

(22) Calzone, L., Chabrier-Rivier, N., Fages, F., and Soliman, S. (2006) *Transactions on Computational Systems Biology VI*; Lecture Notes in Computer Science; Vol. *4220*, pp 68−94, Springer, Berlin Heidelberg.

(23) Gong, H., Zuliani, P., and M. Clarke, E. (2013) Model checking a synchronous diabetes-cancer logical network. *Curr. Bioinf. 8*, 9−15.

(24) Konur, S., Gheorghe, M., Dragomir, C., Ipate, F., and Krasnogor, N. (2014) Conventional verification for unconventional computing: A genetic XOR gate example. *Fund. Informaticae*, Accepted.

(25) Sanassy, D., Fellermann, H., Krasnogor, N., Konur, S., Mierla, L., Gheorghe, M., Ladroue, C., and Kalvala, S. (2014) Modelling and stochastic simulation of synthetic biological Boolean gates. *16th IEEE Int. Conf. of High Performance Comput. Commun.*, Accepted.

(26) Holzmann, G. J. (1997) The model checker SPIN. *IEEE Trans. Software Eng. 23*, 275−295.

(27) Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R., and Tacchella, A. (2002) NuSMV Version 2: An OpenSource tool for symbolic model checking. *Proc. Int. Conf. Computer-Aided Verification (CAV 2002) 2404*, 359−364.

(28) Hinton, A., Kwiatkowska, M., Norman, G., and Parker, D. (2006) A tool for automatic verification of probabilistic systems. *Proc. TACAS 3920*, 441−444.

(29) Hansson, H., and Jonsson, B. (1994) A logic for reasoning about time and reliability. *Formal Aspects Comput. 6*, 102−111.

(30) Baier, C., Haverkort, B., Hermanns, H., and Katoen, J. (2003) Model-checking algorithms for continuous-time Markov chains. *IEEE Trans. Software Eng. 29*, 524−541.

(31) Younes, H. L. S., and Simmons, R. G. (2002) *Computer-Aided Verification*; Lecture Notes in Computer Science; Vol. *2404*; pp 223−235, Springer, Berlin Heidelberg.

(32) Donaldson, R. and Gilbert, D. (2008) *A Monte Carlo Model Checker for Probabilistic LTL with Numerical Constraints*; University of Glasgow, Glasgow, U.K.

(33) Bernardini, F., Gheorghe, M., and Krasnogor, N. (2007) Quorum sensing P systems. *Theoretical Computer Science 371*, 20−33.

(34) Saeidi, N., Arshath, M., Chang, M. W., and Poh, C. L. (2013) Characterization of a quorum sensing device for synthetic biology

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92

design: Experimental and modeling validation. *Chem. Eng. Sci. 103*, 91−99.
(35) Basu, S., Mehreja, R., Thiberge, S., Chen, M.-T., and Weiss, R. (2004) Spatiotemporal control of gene expression with pulse-generating networks. *Proc. Natl. Acad. Sci. U.S.A. 101*, 6355−6360.
(36) Pulse generator. Available online: http://www.infobiotic.org/models/pulseGenerator/pulseGenerator.html (accessed August 9, 2014).
(37) Quorum sensing. Available online: http://staffwww.dcs.shef.ac.uk/people/s.konur/models/qs (accessed August 9, 2014).

DOI: 10.1021/sb500134w
ACS Synth. Biol. 2015, 4, 83−92