Molecular
BioSystems

Cite this: DOI: 10.1039/c0mb00253d

www.rsc.org/molecularbiosystems

PAPER

Online model checking approach based parameter estimation to a neuronal fate decision simulation model in *Caenorhabditis elegans* with hybrid functional Petri net with extension†

Chen Li,[‡][a] Masao Nagasaki,[‡][*][a] Chuan Hock Koh[abc] and Satoru Miyano[a]

Received 29th October 2010, Accepted 2nd February 2011
DOI: 10.1039/c0mb00253d

Mathematical modeling and simulation studies are playing an increasingly important role in helping researchers elucidate how living organisms function in cells. In systems biology, researchers typically tune many parameters manually to achieve simulation results that are consistent with biological knowledge. This severely limits the size and complexity of simulation models built. In order to break this limitation, we propose a computational framework to automatically estimate kinetic parameters for a given network structure. We utilized an online (on-the-fly) model checking technique (which saves resources compared to the offline approach), with a quantitative modeling and simulation architecture named hybrid functional Petri net with extension (HFPNe). We demonstrate the applicability of this framework by the analysis of the underlying model for the neuronal cell fate decision model (ASE fate model) in *Caenorhabditis elegans*. First, we built a quantitative ASE fate model containing 3327 components emulating nine genetic conditions. Then, using our developed efficient online model checker, MIRACH 1.0, together with parameter estimation, we ran 20-million simulation runs, and were able to locate 57 parameter sets for 23 parameters in the model that are consistent with 45 biological rules extracted from published biological articles without much manual intervention. To evaluate the robustness of these 57 parameter sets, we run another 20 million simulation runs using different magnitudes of noise. Our simulation results concluded that among these models, one model is the most reasonable and robust simulation model owing to the high stability against these stochastic noises. Our simulation results provide interesting biological findings which could be used for future wet-lab experiments.

Introduction

Mathematical modeling and simulation studies are playing an increasingly important role in helping researchers elucidate how living organisms function in cells. So far, many formal description methods on biological pathway modeling have been made.[1–4] Among them, Petri net and its related concepts have been successfully applied in modeling a wide variety of biological pathways (*i.e.* metabolic,[5–7] signal transduction,[8–13] gene regulatory networks,[14,15] and cell–cell interactions[16]),[17,18] and have succeeded in reproducing consistent time-series profiles of biological substances such as the concentrations of mRNA and protein by means of computer simulations.

Simulation studies on biological pathways provide great insight into the understanding of complex regulatory mechanisms in cells. Simulation models are usually governed by a series of parameters, *e.g.*, initial values, reaction speeds and threshold values of cellular activities. Commonly, they are carefully tuned by experts to fit the simulated elements with observed *in vivo*/*vitro* experimental results. Numerous trial and error operations are performed until appropriate parameters for simulation are determined. However, this would be extremely tedious if not impossible for simulation studies that are large in scale and complexity.

Inspired by the work of applying a simulation-based model checking approach to complex biological networks,[16] we have developed a computational framework by incorporating the

[a] *Human Genome Center, Institute of Medical Science, University of Tokyo, 4-6-1 Shirokanedai, Minato-ku, Tokyo 108-8639, Japan. E-mail: masao@hgc.jp; Fax: +81 3-5449-5442; Tel: +81 3-5449-5615*
[b] *NUS Graduate School for Integrative Sciences and Engineering, Singapore117597, Singapore*
[c] *School of Computing, National University of Singapore, Computing Drive, Singapore 117417, Singapore*

† Electronic supplementary information (ESI) available: File 1: effects with respect to 21 estimated parameters under different standard deviations for each 57 stochastic models. *x*-Axis shows the standard deviations and *y*-axis denotes the satisfied rule number. File 2: effects with respect to 57 stochastic models under different standard deviations for each 21 estimated parameters. See DOI: 10.1039/c0mb00253d

‡ These authors made equal contributions.

above approach for automatically determining the kinetic parameters in large-scale models instead of the conventional hand tuning method based on biological knowledge. The research of a model checking algorithm for temporal logic was invented about thirty years ago, which is a powerful technique that allows for the automatic verification of system requirements.[19–22] It is receiving more and more attention in studying biological queries of cellular interaction networks.[2,23–29] With the aid of model checking, one can obtain answers to questions such as "Does this reaction always lead to the DNA supercoiling after entering growth-phase?" and "What is the probability that the expression of the gene *g* will be inhibited by the protein *P*?"

The simulation-based model checking approach is developed from the original model checking algorithm, which is based on the idea of the Monte Carlo sampling method, and analyzes approximating results that are a subset of the finite/infinite state space with stochastic simulations.[16,30,31] In ref. 16, the authors have applied this idea to the determination of the vulval precursor cell (VPC) fate in *Caenorhabditis elegans* by means of a large amount (480 000-run) of simulations. They employed two major biological fate determination rules to the VPC fate model with a high-level Petri net and simply calculated the fitting score that is a percentage denoting the coverage of predicted cell fate patterns. No temporal logic theory was taken into account and the parameter assignment of the model was manually decided over an approximate period of six months. This prompts us to develop a more efficient and reliable systematic framework to query dynamic properties of interest as statements with formal verification based on the model checking algorithm, and overcome limitations in the kinetic parameter assignments.

The aim of this article is to propose a novel framework to automatically estimate the kinetic parameters of a given model or a model starting from scratch with the use of a model checking technique incorporated with a quantitative modeling and simulation architecture named hybrid functional Petri net with extension (HFPNe). It is considered to be a great help in facilitating the procedure of the conventional hand tuning before performing the simulations, and is of great value to obtain more accurate, confident executable simulation models, eventually leading to better understanding of biological pathways.

The paper is organized as follows. In Methods, we first present a brief introduction of HFPNe. Then we introduce a temporal logic, *Probabilistic Linear-time Temporal Logic* (PLTL) and the implementation of MIRACH 1.0. We show how biological questions can be translated into PLTL formulas for querying the dynamical properties. In Results and discussion, the applicability of the developed framework is demonstrated by analyzing the quantitative dynamics of a large and complex model of the neuronal cell fate decision model in *C. elegans* (called the ASE model). This model extends the previous HFPNe model[32] by taking into account an additional zinc finger regulator *fozi-1* that functions in the nucleus of ASER (right asymmetric gustatory neuron) to inhibit the expression of the LIM homeobox gene *lim-6*.[33] The software tool Cell Illustrator® is used for modeling and simulation, where the HFPNe was implemented.[34] We demonstrate how our computational framework is employed to estimate the kinetic parameters with observable biological results, *e.g.* measured fate markers' expressions. We estimate different types of parameters (parameters for one initial value, one reaction speed and 21 threshold values) in the ASE simulation model. Furthermore, we evaluate the robustness with respect to the estimated parameters for the ASE simulation model, which simultaneously contains deterministic, stochastic factors. We discuss the simulation results in the presence of noise and select the most robust and appropriate ASE simulation model.

Methods

Fig. 1a illustrates the schematic overview of Methods, along with the short explanation with respect to the application of the ASE model in *C. elegans*. Fig. 1b exhibits the flow chart of the procedures shown in Fig. 1a. In brief, to derive final satisfied parameter sets (rightmost block in Fig. 1a), the HFPNe model and temporal logic rule sets describing biological queries (leftmost two blocks in Fig. 1a) are required as inputs for the model checker. With the repetition of the operations (*i.e.* II–IV in Fig. 1) until the entire rule sets are satisfied, resulting parameter sets can be eventually obtained for simulation and further analysis. In this paper, our method is applied to an updated ASE fate model in *C. elegans*. We extract and translate 45 biological rules specifying dynamical properties with temporal logic (PLTL). By combining the uses of our online model checker (MIRACH 1.0) and a simple estimation technique, 23 kinetic parameters contributing to the regulation of forming alternative cell fates are estimated. Finally, we obtain 57 satisfied parameter sets through 20 million-run simulations (Note: the simulation engine is incorporated in MIRACH 1.0.), and the simulation results of the corresponding models possessing each of the above 57 parameter sets can successfully conform to all 45 biological specifications.

Basic introduction into hybrid functional Petri net with extension (HFPNe)

HFPNe is a mathematical tool for modeling and simulation of biological networks.

HFPNe deals with three types of data—discrete, continuous and generic—and is comprised of three types of elements—*entities*, *processes* and *connectors*—whose symbols are illustrated in Fig. 2a.

- A discrete entity holds an integer number of the content. A continuous entity holds a real number as concentration of a substance such as mRNA and protein. Usually, the value of a discrete or continuous entity is limited to a non-negative one. A generic entity can hold any kind of types including object, *e.g.*, the string of a nucleotide base sequence.
- A discrete process is the same notion as used in the traditional discrete Petri net. A continuous process is used to represent a biological reaction such as transcription and translation, at which the reaction speed is assigned as a parameter. A generic process can deal with any kind of operations (*e.g.*, alternative splicing and frameshifting) to all types of entities. Generic entity and process have been practically applied for modeling and simulating more complicated

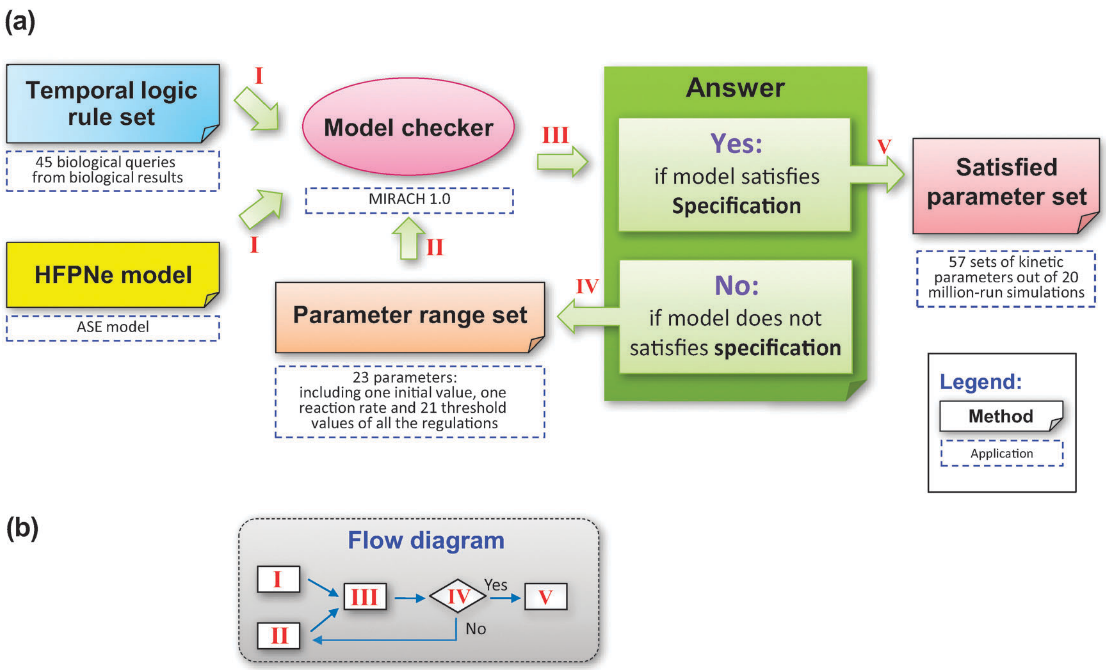

(a)
Temporal logic rule set
45 biological queries from biological results
HFPNe model
ASE model
I
I
Model checker
MIRACH 1.0
II
Parameter range set
23 parameters: including one initial value, one reaction rate and 21 threshold values of all the regulations
III
Answer
Yes: if model satisfies Specification
No: if model does not satisfies specification
IV
V
Satisfied parameter set
57 sets of kinetic parameters out of 20 million-run simulations
Legend:
Method
Application
(b)
Flow diagram
I
II
III
IV
Yes
No
V

Fig. 1 Schematic view of the Methods section. (a) Detailed procedures and the corresponding applications to an ASE fate model; (b) flow diagram of operations shown in (a).

biological processes, *e.g.*, activities of enzymes for a multi-modification protein.

- Connectors are classified into three types: *normal*, *test* and *inhibitory* connectors. Normal connectors connect an entity to a process or *vice versa*. Test or inhibitory connectors represent a condition and are only directed from an entity to a process. Each normal connector from an entity, a test connector, and an inhibitory connector has a threshold by which the parameter assigned to the process at its head is controlled. A normal connector from an entity or a test connector (an inhibitory connector) can participate in activating (repressing) a process at its head, as far as the content of an entity at its tail is over the threshold. For either of the test or inhibitory connectors, no amount is consumed from an entity at its tail. Fig. 2b illustrates above connection rules in HFPNe.

Owing to the versatility of HFPNe, it has been successfully employed to develop and analyze complex biological networks.[13,36] For example, Wu *et al.*[13] applied HFPNe to model and evaluate the dopamine signaling pathway accounting for delays and noise in the system. Hawari and Mohamed-Hussein[36] used HFPNe to model a metabolic pathway, namely the terpenoid biosynthesis pathway. The model simulated metabolite concentration changes over virtual simulation time and observations correlated with known experimental data. For more comprehensive description of HFPNe and its usage, please see Nagasaki *et al.*[34,35]

Formulating biological queries in temporal logic

To verify user queries of a specified property by means of model checking, several approaches of temporal logics have been employed for model checking in biological pathway models. The approaches are not restricted to the application of basic Computation Tree Logic[23,24] or Linear-time Temporal Logic (LTL),[29] and other more expressive temporal logics, *e.g.*, Alternative Temporal Logic,[37] Continuous Time Evolution Logic,[12] and Probabilistic Bounded Linear Temporal Logic,[38] are also proposed and applied to diverse biological systems.

In this article, we selected PLTL (*Probabilistic Linear-time Temporal Logic*) for querying dynamic models of cellular networks,[31,39,40] which extends original LTL to a stochastic setting with a probability operator and a filter criterion defining the starting state where the property is satisfied. Delving into further detail, we utilize PLTL on the architecture of HFPNe dealing with both discrete and continuous events. Meanwhile, we had developed a model checker, MIRACH 1.0, which implements PLTL and its new extension, namely PLTLs (*Probabilistic Linear-time Temporal Logic with Statistics*).[41] Our model checker concurrently supports any model written in commonly-used formats such as CSML (http://www.csml.org) and SBML (http://sbml.org/). MIRACH is integrated with a simulation engine, enabling efficient online (on-the-fly) checking, and producing a definite and reliable (statistically backed) result. Online model checking performs model checking during the simulation run,[12,42] whereas the offline model checking approach needs to complete the simulation runs before the checking, therefore the offline approach would be a waste of CPU resources if the decision of validity or rejection for the simulation run could be determined early in its execution.

Syntax of PLTL. The syntax of PLTL is defined in Table 1, which is used to ask for the probability of user's query *via* a PLTL formulae ψ. In the LTL expression $\phi\{\mathrm{AP}\}$, ϕ will be checked from the state that AP is satisfied rather than from the default initial state, where AP is called *atomic proposition* and takes a boolean domain. PLTL allows (i) LTL expression

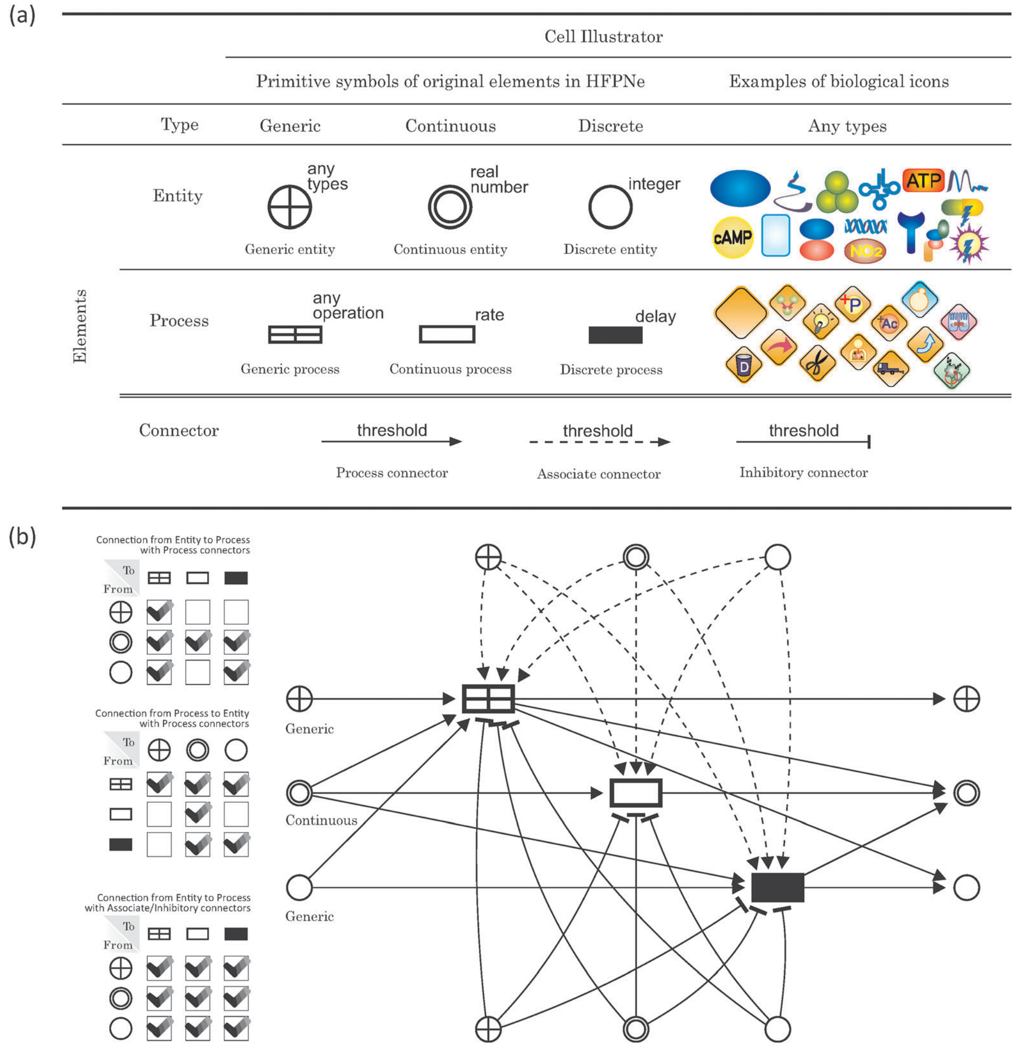


Fig. 2 (a) Basic HFPNe elements and biological icons in Cell Illustrator. (b) Connection rules (left side) and the corresponding network (right side) in HFPNe. For instance, for the uppermost block labeled with "Connection from Entity to Process with Process connectors", the check-mark denotes the availability connected from the corresponding entities to processes, *e.g.*, only the generic process can be selected as the output connected from a generic entity with a process connector.

Table 1 Syntax of PLTL

ψ	$::=$	$\mathbf{P}_{\trianglelefteq x}(LTL) \mid \mathbf{P}_{=?}(LTL) \mid LTL$
LTL	$::=$	$\phi\{AP\} \mid \phi$
ϕ	$::=$	$\mathbf{X}\ \phi \mid \mathbf{G}\ \phi \mid \mathbf{F}\ \phi \mid \phi\ \mathbf{U}\ \phi \mid \phi\ \mathbf{R}\ \phi \mid \phi \mid \phi\ \&\&\ \phi \mid \phi \Vert \phi \mid \phi \Rightarrow \phi \mid \mathrm{AP}$
AP	$::=$	$value\ comp\ value \mid value_{boolean}$
$Value$	$::=$	$value\ op\ value \mid [variableName] \mid Function_{numeric} \mid Integer \mid Real$
$Value_{boolean}$	$::=$	$true \mid false \mid Function_{boolean}$
$Comp$	$::=$	$= = \mid != \mid \geq \mid > \mid < \mid \leq$
Op	$::=$	$+ \mid - \mid * \mid / \mid \hat{}$,
with $\trianglelefteq \in \{<, \leq, >, \geq\}$, $x \in [0, 1]$.		

to contain temporal operators, *i.e.*, **X**, **F**, **G**, **U**, **R**. Five temporal operators are used to describe the sequencing of the states along the execution; and (ii) the usage of ψ without probabilistic operators (*i.e.* simply in the form of LTL), which is useful when the model is deterministic.

Semantics of PLTL. The semantics of PLTL is defined over the finite sets of finite paths through system's state space, obtained by repeated simulation runs of HFPNe models. The PLTL formula is built upon two components: probabilistic operator and property *LTL*. For each simulation run, the LTL expression is evaluated to a boolean truth value, and the probability of the LTL statement holding true is calculated based on the whole set of simulation results.

For the probability operator components, there are two distinct operators: (i) $\mathbf{P}_{\trianglelefteq x}$ is any inequality comparison of the

Table 2 Semantics of temporal operators

Operator	Meaning	Explanation
X ϕ	Next time	ϕ must be true at the next time point.
G ϕ	Globally	ϕ must always be true.
F ϕ	Finally	ϕ must be true at least once.
ϕ_1 **U** ϕ_2	Until	ϕ_1 must be true until ϕ_2 becomes true; ϕ_2 must become true eventually.
ϕ_1 **R** ϕ_2	Release	ϕ_2 must be true until and including the time point ϕ_1 becomes true; if ϕ_2 never true, ϕ_1 must always be true.

probability of the property LTL holding true, for example $\mathbf{P}_{\geq 0.5}$ (LTL); and (ii) $\mathbf{P}_{=?}$ returns the value of the probability of the property holding true. The semantics of the temporal logic operators are described in Table 2. Concentrations of biochemical species in the model are denoted by [*variableName*]. A special variable, [*time*], stands for the simulation time.

Due to the ability of PLTL, it is possible to define functions of two different natures: functions that return a real number and functions that return a boolean value. An example of the real number function is *d*([*variableName*]) which returns the subtracted value of [*variableName*] between time i and $i - 1$. Note that, *d*([*variableName*]) equals zero at time point zero. One example of a boolean function is *similarAbsolute*(*value a*, *value b*, *value* ε), which returns true if $|a - b| \leq \varepsilon$ or else it returns false. Please see MIRACH's documentation for details and examples on usage of implemented functions (http://sourceforge.net/projects/mirach/).

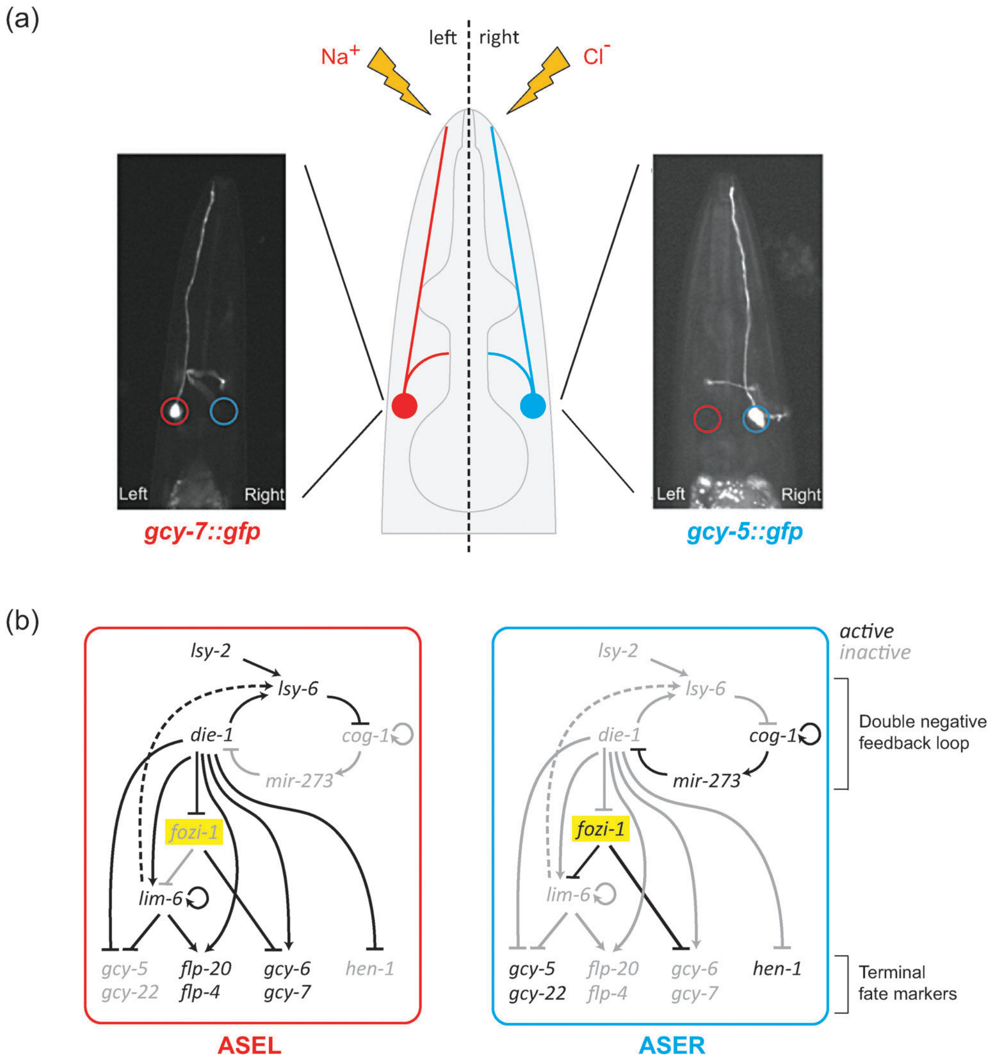


Fig. 3 Summary of the regulatory interactions that determine the ASEL/ASER fate. (a) Two ASE neurons. ASE senses different ions and expresses distinct ASEL/ASER-specific terminal fate markers, encoded by *gcy* and *flp* family genes. Photomicrographs of ASEL/ASER-specific *gcy* gene expressions in wild type are adapted from Hobert, 2006, Cold Spring Harbor Laboratory Press ©2006. (b) Biological diagram of the ASE neuron fate decision pathway which takes into account an additional regulator, *fozi-1* (highlighted) and *fozi-1* related regulations. Broken line denotes a partially penetrant defect in maintaining the left/right asymmetric expression of loop component.[33] Genes in the inactive or active state are shown in grey or black, respectively. Four regulatory factors *lsy-6*, *cog-1*, *die-1* and *mir-273* form a double-negative feedback loop (DNFL). The expressions of *flp-20*/*flp-4* and *gcy-6*/*gcy-7* are ASEL-specific terminal fate markers, while the expressions of *gcy-5*/*gcy-22* and *hen-1* are used as ASER fate markers.

Application and results

To confirm the applicability of our developed framework, we chose the HFPNe model of neuronal fate decision mechanisms in *C. elegans* for the analysis. Fig. 3 illustrates the summary and biological diagram of the mechanisms by considering a new transcriptional factor, *fozi-1* (highlighted in Fig. 3b) and complicated regulations mediated by *fozi-1*.

Biological background and HFPNe modeling of an ASEL/R cell fate regulatory network

Two gustatory neurons in *C. elegans*, "ASE left" (ASEL) and "ASE right" (ASER) are morphologically bilaterally symmetric, but physically asymmetric in function and in the expression of distinct ASEL/ASER-specific cell fate markers, including specific subsets of guanylyl cyclase receptors, encoded by GCY genes (*e.g. gcy-5* and *gcy-7*), and FMRFamide-type neuropeptides, encoded by FLP genes (*e.g. flp-4*). In adult animals, the differences between cell fate markers are used to discriminate ASEL or ASER cells. That is, *gcy-6* and *flp-4* are stereotypically expressed in the ASEL cell, whereas *gcy-5* is expressed only in the ASER cell as shown in Fig. 3. The left/right asymmetric fates develop from a precursor state in which both ASE neurons have equivalent potentials to adopt alternative cell fates.[32,43,44] The ASE cell fate decision mechanism between two alternative neuronal fates is controlled by a complex gene regulatory network composed of microRNAs (miRNAs) (*e.g. lsy-6* and *mir-273*) and transcription factors (*e.g. cog-1*, *lim-6* and *die-1*). This mechanism diversifies neuronal subclass specification (*i.e.* wild type and classes I–IV, see Johnston *et al.*, 2006, Fig. 4C) in the nervous system of the nematode *C. elegans*.

In the ASE cell fate decision mechanism, a double-negative feedback loop (DNFL) (Fig. 3b) constituted by the regulatory factors *lsy-6*, *cog-1*, *die-1* and *mir-273* plays an important role to provide the establishment and stabilization of the bi-stable ASE system. In 2006, Johnston *et al.* isolated a mutant, *fozi-1*, and it is characterized by de-repression of the ASEL fate in ASER *via* genetic experiments. *fozi-1* codes for a protein, containing two Zn fingers and a single FH2 domain, that functions in the nucleus of ASER to inhibit the expression of LIM homeobox gene *lim-6*.[33] In other words, *fozi-1* genetically interacts with a series of transcription factors and miRNAs to repress expression of ASEL-specific effector genes in ASER to adopt the terminally stable ASER cell fate.

In previous work we developed an HFPNe model (termed "**Model2**") that we extend here by updating the regulatory interactions mediated by *fozi-1*. Fig. 4 exhibits our HFPNe model of the ASE fate decision pathway in wild-type depicted in Fig. 3, and the elements of HFPNe in Fig. 4 are changed to biological icons in Cell Illustrator (on the right side of Fig. 2a). Although these changes have no effect on mathematical meaning, it is helpful for biologists to easily make and understand the pathway models. Table 3 summarizes the biological interpretation and references of each reaction used in this study. Fig. 5 shows the schematic representation of the whole HFPNe model that emulates nine genetic conditions including not only wild-type but also the combinations of five mutants, *fozi-1*(*cc607*), *die-1*(*ot26*), *lsy-6*(*ot71*), *cog-1*(*sy607*) and *lim-6*(*nr2073*). The whole ASE model is composed of 474 entities, 1026 processes and 1620 connectors (totally 3327 components). The HFPNe model and related data files of analyzing the ASE fate decision mechanism in *C. elegans* are available at the website (http://www.csml.org/models/csml-models/ase-cell-fate-simulation/ASE2010/).

Parameter estimation coupled with the online model checking approach

In the original model built by Saito *et al.*[32] (*i.e.* **Model1**), no kinetic parameters have been documented and measured in any literature. The kinetic parameters are thus assigned according to two simple principles: (i) the same speed $m_x \times 0.1$ (m_x indicates the concentration of the substance) is used for the regulations (*i.e.*, translation, transcription of mRNA, nuclear import and export); (ii) the threshold value of the inhibitory connector is 0.1. Note that the transcription speed of miRNA is tuned to $m_x \times 0.01$, and the threshold of inhibitory connector from the miRNA is tuned to 0.125. We here assign the kinetic parameters to updated processes and inhibitory connectors in **Model2** complying with the above principles. In order to check constructed models (**Model1** and **Model2**) for consistency and correctness before starting further analyses, in the next step we focus on model validation by using the model checking approach. We thus establish a temporal logic rule set for verification.

The temporal logic rule set includes 45 rules in total, extracted from *in vivo* results with respect to the dynamic behaviour of the ASE fate specification pathway. Forty-five rules are formulated with PLTL subsequently as shown in Table 4. For instance, as shown in Fig. 6a, Johnston *et al.*[43] have suggested that "In wild type, the expression pattern of *mir-273* gene adopts $L > R$, $L = R$ or $L < R$". This biological fact thus is translated into a PLTL syntax as given in Fig. 6b. Note that the variable [*thres_l*] is used to denote a threshold value that is close to zero; whereas [*thres_h*] is used to discriminate the expression differences between ASEL and ASER which is unambiguous when biologists say "$L > R$" or "$L < R$" in their community.

Table 5 shows the model checking results for **Model1** and **Model2**. We can see that there are two FALSEs and 14 SKIPs of **Model1**. That is, **Model1** cannot conform to the biological criteria described in Rules 14 and 15, and SKIP occurred because no information of the new factor *fozi-1* was taken into account. This implies that **Model1** is relatively well tuned in the previous work. However, after updating the information of *fozi-1* related interactions, **Model2** can only be satisfied with 33 rules out of 45 (about 73% precision) with the same parameter assignment principles defined in **Model1**. It is obvious that current parameter assignment principles are not suitable for further analysis and behaviour predictions and should be improved to increase the correctness and confidence of the model. Therefore, a new systematic approach is needed to find out appropriate parameter assignments where the HFPNe model will satisfy 45 specified rules.

In this study, 23 kinetic parameters involving one initial value, one reaction rate and 21 threshold values of the regulatory (*i.e.* inhibitory and associate) interactions are estimated using

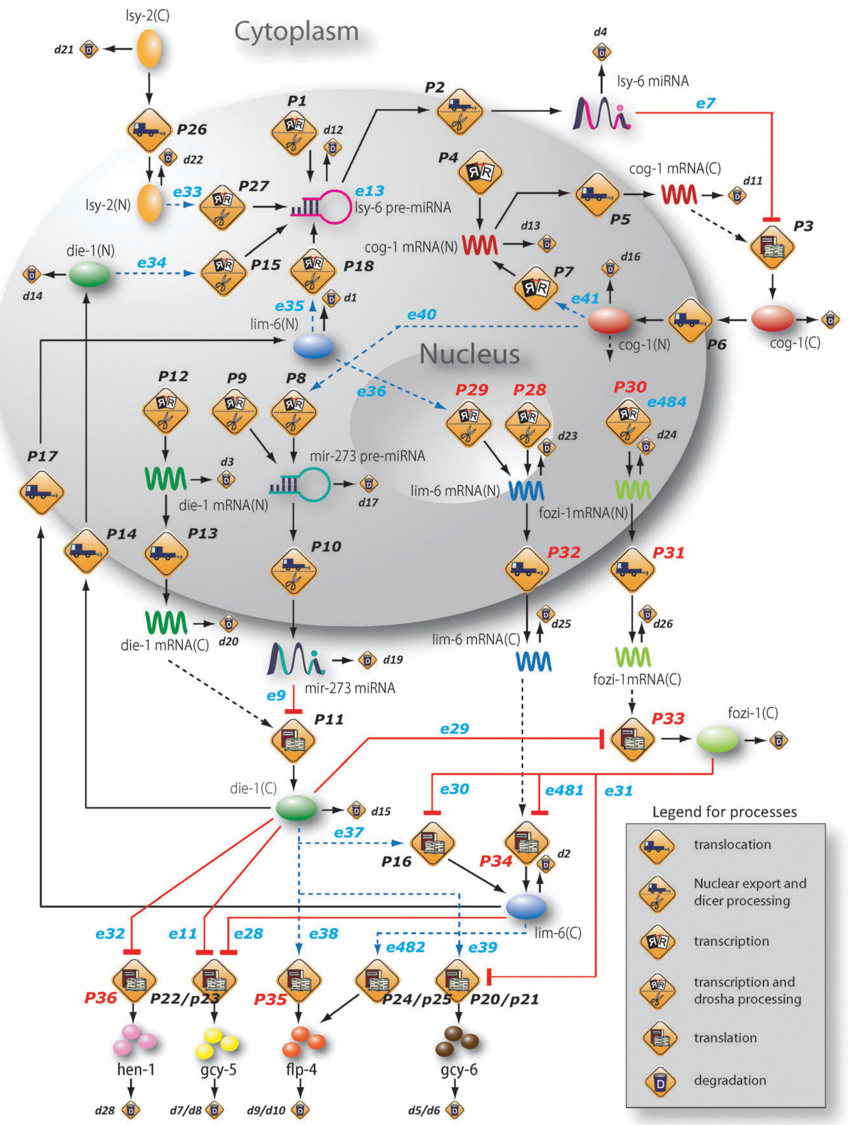


Fig. 4 The HFPNe model of the ASE fate decision pathway in wild-type depicted in Fig. 3b. Biological meanings of processes ***P1***,. . ., ***P36*** are summarized in Table 3. An additional label (C) or (N) is attached at the end of a substance name, when it happens to distinguish the location of the substance in the cytoplasm or the nucleus. The entities' IDs used for parameter estimation are indicated in blue.

uniform distribution in the setting for range values summarized in Table 6. Note that the scope (the column "Range values" in Table 6) of the parameters for estimation is narrowed down by repeating the operations of estimation and model checking. It is observed that the models with parameter values beyond the value ranges stated in Table 6, have high possibility of violating some of these 45 rules.

Given reduced range values, we have tried matured (20 000 000 run) simulation experiments for the parameter estimations with the same model structure (*i.e.*, **Model2** with the remaining kinetic parameters), the same procedures and the same computational environment in order to find out suitable model that could satisfy all 45 specifications.

Model selection with stochastic noise

Using the method mentioned in previous sections, we performed the parameter estimations coupled with online model checking for the ASE fate decision model. We obtained 57 kinetic parameter sets out of 20 million-run simulations whose respective model (summarized to "**Model3**") can satisfy all the 45 rules. Although these simulation models can meet all the specified biological criteria, we are interested to find an adaptive model that is robust enough to resist the effects of stochastic noise induced by the intrinsic stochasticity and external perturbations.

We then carry out model selection based on the system's robustness under disturbances arising from noise. On account of the stochasticity in a dynamic system, the software Cell Illustrator facilitates the robustness analysis which can simultaneously contain deterministic and stochastic components. We converted the obtained 57 deterministic models to stochastic ones by adding the function of log-normal distribution as a system noise equipped in Cell Illustrator. Note that, 57 deterministic models are named as Model3_1, Model3_2,. . ., Model3_57, respectively. Log-normal

Table 3 Biological interpretation of each reaction in Fig. 4 based on the literature. The processes {*p1*, *p2*,. . .,*p27*} are adapted from the original model (Saito *et al.*[32] called "Model1"). Note that, the names of adapted processes are changed from "***T***" to "***p***" to make the paper to be more self-contained. Nine *fozi-1*-related interactions are assigned to the processes {*p28*, *p29*,. . ., *p36*}, which are indicated by boldface. {*d1*, *d2*,. . .,*d28*} represents natural degradations of attached substances

Process			
Name	Wet experiment results published in the literature	Reaction type	Ref.
p1	Transcription of the *lsy-6* gene, produces *lsy-6* pre-miRNA, and Drosha processing yields the *lsy-6* pre-miRNA	Transcription/Drosha processing	32
p2	The *lsy-6* pre-miRNA is exported from the nucleus to the cytoplasm by exportin-5 and processed by the dicer (*lsy-6* miRNA) to form miRNA	Nuclear export/Dicer processing	
p3	*cog-1* mRNA(C) is translated to *cog-1*(C) under the suppression by *lsy-6* miRNA (within RISC)	Translation/microRNA inhibition	
p4	Transcription of *cog-1* gene yields *cog-1* mRNA	Transcription	
p5	*cog-1* mRNA(N) is exported from the nucleus to the cytoplasm (*cog-1* mRNA (C))	Nuclear export	
p6	COG-1(C) is imported from the cytoplasm to the nucleus (COG-1(N)).	Nuclear import	
p7	COG-1(N) activates transcription of the *cog-1* gene, producing *cog-1* mRNA	Transcription	
p8	COG-1(N) activates the transcription of the *mir-273* gene, producing the *mir-273* pre-miRNA, and Drosha processing leads to the production of the *mir-273* pre-miRNA	Transcription/Drosha processing	
p9	Transcription of the *mir-273* gene yields the *mir-273* pre-miRNA, and Drosha processing produces the *mir-273* pre-miRNA	Transcription/Drosha processing	
p10	*mir-273* pre-miRNA is exported from the nucleus to the cytoplasm by exportin-5 and processed by the dicer (*mir-273* miRNA) to yield miRNA	Nuclear export/dicer processing	
p11	*die-1* mRNA(C) is translated to DIE-1(C) under suppression by the *mir-273* miRNA (within RISC)	Translation/microRNA inhibition	
p12	Transcription of *die-1* gene leads to the production of *die-1* mRNA.	Transcription	
p13	*die-1* mRNA(N) is exported from the nucleus to the cytoplasm (*die-1* mRNA(C))	Nuclear export	
p14	*die-1*(C) is imported from the cytoplasm to the nucleus (*die-1*(N))	Nuclear import	
p15	*die-1*(N) activates the transcription of the *lsy-6* gene, producing the *lsy-6* pre-miRNA, and Drosha processing leads to the production of the *lsy-6* pre-miRNA	Transcription/Drosha processing	
p16	The expression of *lim-6*(C) is activated by *die-1*(C) and suppressed by *cog-1*(C)	Expression	
p17	LIM-6(C) is imported from the cytoplasm to the nucleus (LIM-6(N))	Nuclear import	
p18	LIM-6(N) activates the transcription of *lsy-6* gene, producing *lsy-6* pre-miRNA, and Drosha processing leads to the production of *lsy-6* pre-miRNA	Transcription/Drosha processing	
p19	LIM-6(N) activates the transcription of *die-1* gene, producing *die-1* mRNA	Transcription	
p20	The expression of *gcy-7* is activated by DIE-1 and suppressed by COG-1	Expression	
p21	The expression of *gcy-6* is activated by DIE-1 and suppressed by COG-1	Expression	
p22/*p23*	LIM-6(C) suppresses the expression of *gcy-5*/*gcy-22* gene	Expression	
p24/*p25*	LIM-6(C) activates the expression of *flp-4*/*flp-20* gene	Expression	
p26	Protein LSY-2(C) is imported from the cytoplasm to the nucleus (LSY-2(N)).	Expression	
p27	LSY-2(N) activates transcription of gene *lsy-6*, producing *lsy-6* pre-miRNA, and the Drosha processes to produce *lsy-6* pre-miRNA	Nuclear import	
p28	Transcription of *lim-6* gene which yields *lim-6* mRNA	Transcription/Drosha processing	43,44
p29	LIM-6(N) activates the transcription of *lim-6* gene which produces *lim-6* mRNA	Transcription	
p30	Transcription of *fozi-1* gene which yields *fozi-1* mRNA	Transcription	
p31	*fozi-1* mRNA(N) is exported from the nucleus to the cytoplasm (*fozi-1* mRNA (C))	Transcription	
p32	*lim-6* mRNA(N) is exported from the nucleus to the cytoplasm (*lim-6* mRNA (C))	Nuclear export	
p33	*fozi-1* mRNA(C) is translated to FOZI-1(C) which is repressed by DIE-1(C)	Nuclear export	
p34	*lim-6* mRNA(C) is translated to LIM-6(C) which is repressed by FOZI-1(C)	Translation/inhibition	
p35	DIE-1(C) activates the expression of *flp-4*/*flp-20* gene	Translation/inhibition	
p36	DIE-1(C) represses the translation of *hen-1* gene	Translation/inhibition	
d1–d28	Natural degradation of attached substances	Degradation	

distribution is represented by *LSMass*(*arg1*,*arg2*), where *arg1* stands for normal reaction speed without noise, and *arg2* is the standard deviation representing the strength of the noise. For each estimated parameter (except *e477* and *e478*, 21 parameters in total) summarized in the first column of Table 6, five standard deviations: 0.1, 0.2, 0.3, 0.4, and 0.5 are respectively utilized in order to investigate and evaluate the effects of the system in terms of noise magnitude. We assigned the log-normal distribution with five standard deviations to each of the above 21 parameters in each 57 models. That is, the number of total models for evaluation is 5 × 21 × 57 = 5985. Moreover, 10 000-run simulation experiments for every 5985 stochastic models are executed to investigate the behavioural variations against the 45 biological rules.

Results

File 1 (ESI†) shows the simulation results concerning the effects with respect to 21 estimated parameters under different standard deviations for 57 models by investigating 5985 stochastic models. File 2 (ESI†) exhibits the results evaluating the fluctuations from the viewpoint of 21 estimated parameters. From the simulation results, we draw the following conclusions: (Here we only show the results of Model3_14 in Fig. 7 due to space constraints.)

(i) Model3_14 appears to be the most robust model compared to the other 56 candidate models. Under different noise strength levels, Model3_14 exhibits good robustness to the

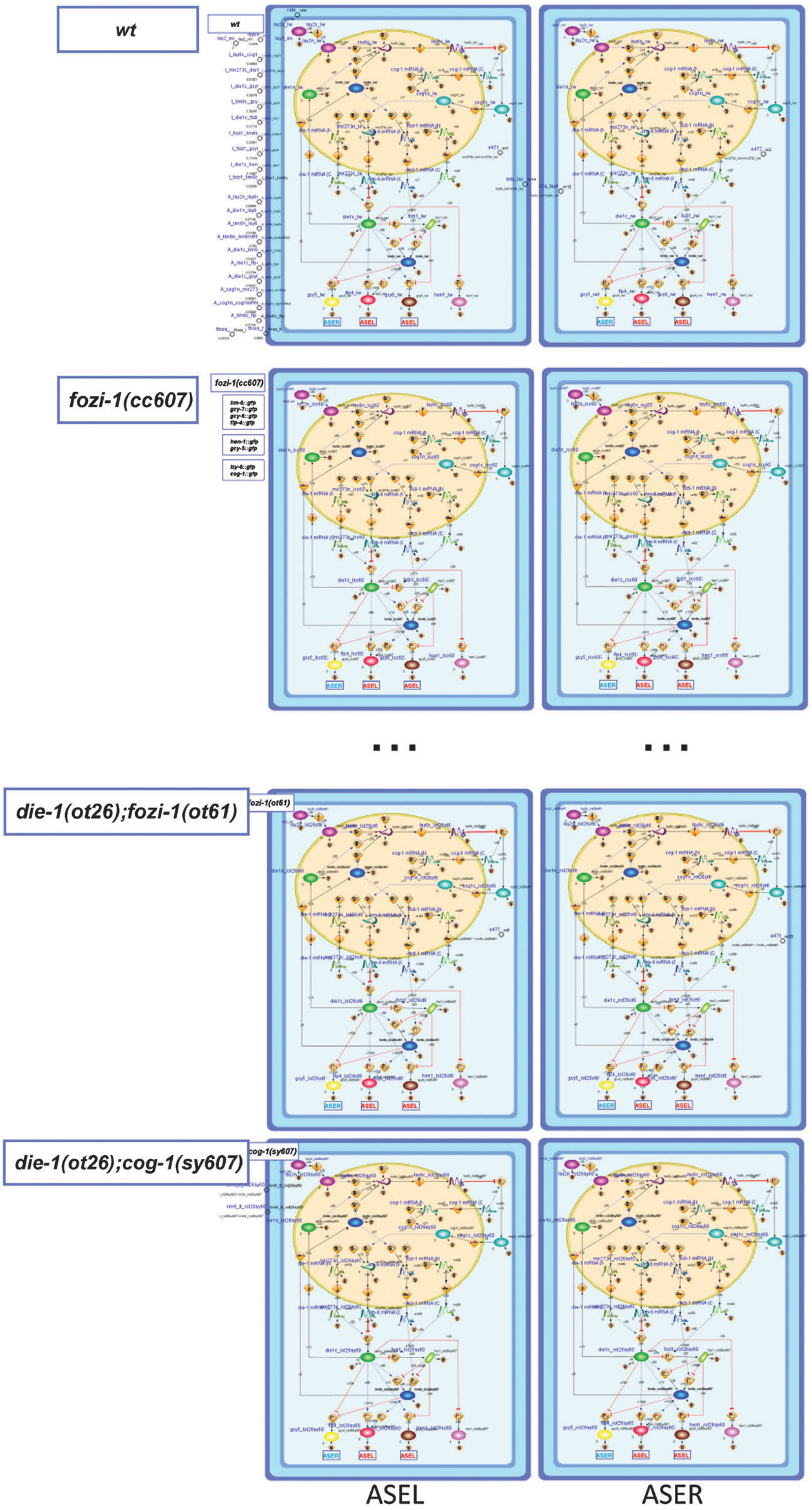


Fig. 5 Schematic representation of the whole HFPNe based ASE fate model underlying alternative fate decision mechanisms involving nine genetic conditions that are shown on the left side. The figure with high resolution and the model are available at the website (http://www.csml.org/models/csml-models/ase-cell-fate-simulation/ASE2010/). As for the label of each substance, the format is defined in the following order: (1) the name of the substance, *e.g.*, *lsy-2*, *die-1* and *cog-1*; (2) "c" or "n" is used to indicate the location of the substance. "c" stands for cytoplasm whereas "n" stands for nucleus; (3) "l" or "r" denotes ASEL or ASER neuron; (4) "wt" represents genetic condition "wild-type". Therefore, the label "die1c_rsy607" could be read as "In *cog-1*(*sy607*) mutant animal, the protein encoded by the *die-1* gene in the cytoplasm in the ASER neuron".

Table 4 Translation of extracted biological evidences into PLTL rules. 0 = 0: no expression; $L > 0$: exclusive expression in ASEL; $L > R$: expression in ASEL is stronger than in ASER; $L = R$ equal expression in ASEL and ASER; $L < R$: expression in ASER is stronger than in ASEL; $0 < R$: exclusive expression in ASER. [GENEc_lwt]: in wild type, expression of GENE in the cytoplasm in the ASEL neuron; for instance, [die1n_rot71] represents the expression of *die-1* gene in the nucleus in the ASER neuron of *lsy-6*(*ot71*) mutants

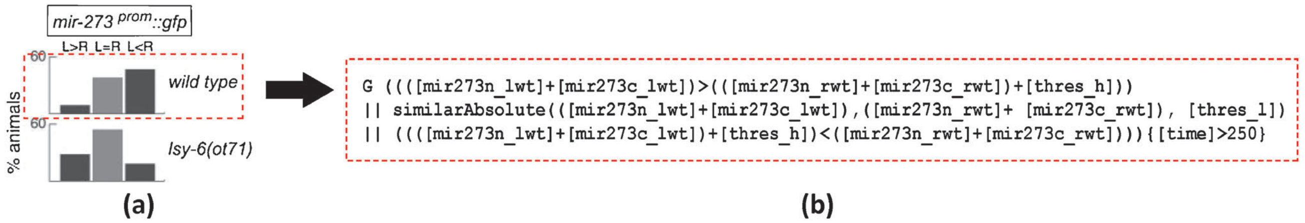
	Rules	Biological evidences
No.	Translation	
Rule 1	LSY-2 in the nucleus will never increase if it once begins to rise and fall. (d([lsy2n_lwt]) ≥ 0) U (G (d([lsy2n_lwt]) ≤ 0))	ref. 32
Rule 2	There exist no expressions of ASEL/ASER-specific reporter genes (*i.e.*, *gcy-5*, *flp-4*,*gcy-6*, *hen-1*) in ASER neuron X (similarAbsolute([gcy5_rwt], 0, [thres_h]) && similarAbsolute([flp4_rwt], 0, [thres_h]) && similarAbsolute([gcy6_rwt], 0, [thres_h]) && similarAbsolute([hen1_rwt], 0, [thres_h])) {[time] = = 0}	
Rule 3	LSY-2 in the cytoplasm keeps decreasing. G (d ([lsy2c_lwt]) ≤ 0)	
Rule 4	After 250 time point, when the concentration of DIE-1 is greater than that of COG-1 in the nucleus, the concentration of LSY-6 will be more than that of MIR-273 in the cytoplasm. F ((([die1n_lwt] > [cog1n_lwt]) {[time] > 250}) && (([lsy6c_lwt] > [mir273c_lwt]) {[time] > 250}))	
Rule 5	The concentration of LIM-6 in the nucleus when DIE-1 is greater than COG-1 in the nucleus is greater than that of LIM-6 in the nucleus when DIE-1 is less than COG-1 in the nucleus. F ([lim6n_lwt] > [lim6n_rwt]) {([time] > 250) && ([die1n_lwt] > [cog1n_lwt]) && ([die1n_rwt] < [cog1n_rwt])}	
Rule 6	If the concentration of LSY-2 in the cytoplasm of the initial state is larger than 0.5, the concentrations of GCY-7 and FLP-4 are greater than those of GCY-5 and HEN-1 after 250 time point. (([lsy2c_lwt] > 0.5) && [time] = = 0) = > F (([gcy6_lwt] > [gcy5_lwt]) && ([gcy6_lwt] > [hen1_lwt]) && ([flp4_lwt] > [gcy5_lwt]) && ([flp4_lwt] > [hen1_lwt])) {[time] > 250}	
Rule 7	In wild type, the expression pattern of *mir-273* gene adopts L > R, L = R or L < R G ((([mir273n_lwt] + [mir273c_lwt]) > (([mir273n_rwt] + [mir273c_rwt]) + [thres_h])) ∥ similarAbsolute(([mir273n_lwt] + [mir273c_lwt]), ([mir273n_rwt] + [mir273c_rwt]), [thres_l]) ∥ ((([mir273n_lwt] + [mir273c_lwt]) + [thres_h]) < ([mir273n_rwt] + [mir273c_rwt]))) {[time] > 250}	Fig. 2A of ref. 43
Rule 8	In *lsy-6*(*ot71*) mutants, the expression pattern of *mir-273* gene adopts $L > R$, $L = R$ or $L < R$ G ((([mir273n_lot71] + [mir273c_lot71]) > (([mir273n_rot71] + [mir273c_rot71]) + [thres_h])) ∥ similarAbsolute(([mir273n_lot71] + [mir273c_lot71]), ([mir273n_rot71] + [mir273c_rot71]), [thres_l]) ∥ ((([mir273n_lot71] + [mir273c_lot71]) + [thres_h]) < ([mir273n_rot71] + [mir273c_rot71]))) {[time] > 250}	
Rule 9	In *cog-1*(*sy607*) mutants, the expression pattern of *mir-273* gene adopts $L > R$, $L = R$ or $L < R$ G ((([mir273n_lsy607] + [mir273c_lsy607]) > (([mir273n_rsy607] + [mir273c_rsy607]) + [thres_h])) ∥ similarAbsolute(([mir273n_lsy607] + [mir273c_lsy607]), ([mir273n_rsy607] + [mir273c_rsy607]), [thres_l]) ∥ ((([mir273n_lsy607] + [mir273c_lsy607]) + [thres_h]) < ([mir273n_rsy607] + [mir273c_rsy607]))) {[time] > 250}	
Rule 10	In the wild type, the expression pattern of *die-1* gene adopts $L > R$, $L = R$ or $L < R$ G ((([die1n_lwt] + [die1c_lwt]) > (([die1n_rwt] + [die1c_rwt]) + [thres_h])) ∥ (similarAbsolute(([die1n_lwt] + [die1c_lwt]), ([die1n_rwt] + [die1c_rwt]), [thres_l])) ∥ ((([die1n_lwt] + [die1c_lwt]) + [thres_h]) < ([die1n_rwt] + [die1c_rwt]))) {[time] > 250}	Fig. 2B of ref. 43
Rule 11	In *die-1*(*ot26*) mutants, the expression pattern of *die-1* gene (*ceh-36*^{prom}*::gfp::die-1 3′UTR*) adopts $L > R$, $L = R$ or $L < R$ G ((([die1n_lot26] + [die1c_lot26]) > (([die1n_rot26] + [die1c_rot26]) + [thres_h])) ∥ (similarAbsolute(([die1n_lot26] + [die1c_lot26]), ([die1n_rot26] + [die1c_rot26]), [thres_l])) ∥ ((([die1n_lot26] + [die1c_lot26]) + [thres_h]) < ([die1n_rot26] + [die1c_rot26]))) {[time] > 250}	
Rule 12	In *lsy-6*(*ot71*) mutants, the expression pattern of *die-1* gene (*ceh-36*^{prom}*::gfp::die-1 3′UTR*) adopts $L > R$, $L = R$ or $L < R$ G ((([die1n_lot71] + [die1c_lot71]) > (([die1n_rot71] + [die1c_rot71]) + [thres_h])) ∥ (similarAbsolute(([die1n_lot71] + [die1c_lot71]), ([die1n_rot71] + [die1c_rot71]),[thres_l])) ∥ ((([die1n_lot71] + [die1c_lot71]) + [thres_h]) < ([die1n_rot71] + [die1c_rot71]))) {[time] > 250}	
Rule 13	In *cog-1*(*sy607*) mutants, the expression pattern of *die-1* gene (*ceh-36*^{prom}*::gfp::die-1 3′UTR*) adopts $L > R$, $L = R$ or $L < R$ G ((([die1n_lsy607] + [die1c_lsy607]) > (([die1n_rsy607] + [die1c_rsy607]) + [thres_h])) ∥ (similarAbsolute(([die1n_lsy607] + [die1c_lsy607]), ([die1n_rsy607] + [die1c_rsy607]), [thres_l])) ∥ ((([die1n_lsy607] + [die1c_lsy607]) + [thres_h]) < ([die1n_rsy607] + [die1c_rsy607]))){[time] > 250}	
Rule 14	In the wild type, the expression pattern of *lsy-6* gene adopts 0 = 0 or $L > 0$ G ((similarAbsolute(([lsy6n_lwt] + [lsy6c_lwt]), 0, [thres_l]) && similarAbsolute(([lsy6n_rwt] + [lsy6c_rwt]), 0, [thres_l])) ∥ ((([lsy6n_lwt] + [lsy6c_lwt]) > [thres_h]) && similarAbsolute(([lsy6n_rwt] + [lsy6c_rwt]), 0, [thres_l]))) {[time] > 250}	Fig. 2C of ref. 43; Fig. 5D of ref. 33

Table 4 (***continued***)

	Rules	Biological evidences
No.	Translation	
Rule 15	In *die-1*(*ot26*) mutants, the expression pattern of *lsy-6* gene adopts $0=0$ or $L > 0$ G ((similarAbsolute(([lsy6n_lot26] + [lsy6c_lot26]), 0, [thres_l]) && similarAbsolute(([lsy6n_rot26] + [lsy6c_rot26]), 0, [thres_l])) ‖ ((([lsy6n_lot26] + [lsy6c_lot26]) > [thres_h]) && similarAbsolute(([lsy6n_rot26] + [lsy6c_rot26]), 0, [thres_l]))) {[time] > 250}	Fig. 2C of ref. 43
Rule 16	In *lsy-6*(*ot71*) mutants, the expression pattern of *lsy-6* gene is $0=0$ G (similarAbsolute(([lsy6n_lot71] + [lsy6c_lot71]), 0, [thres_l]) && similarAbsolute(([lsy6n_rot71] + [lsy6c_rot71]), 0, [thres_l])){[time] > 250}	
Rule 17	In *cog-1*(*sy607*) mutants, the expression pattern of *lsy-6* gene adopts $L > 0$, $L = R$ or $0 < R$ G (((([lsy6n_lsy607] + [lsy6c_lsy607]) > [thres_h]) && similarAbsolute(([lsy6n_rsy607] + [lsy6c_rsy607]), 0, [thres_l])) ‖ (similarAbsolute(([lsy6n_lsy607] + [lsy6c_lsy607]), ([lsy6n_rsy607] + [lsy6c_rsy607]), [thres_l]) && !(similarAbsolute(([lsy6n_rsy607] + [lsy6c_rsy607]), 0, [thres_l]))) ‖ ((([lsy6n_rsy607] + [lsy6c_rsy607]) > [thres_h]) && similarAbsolute(([lsy6n_lsy607] + [lsy6c_lsy607]), 0, [thres_l]))) {[time] > 250}	
Rule 18	In the wild type, the expression pattern of *lim-6* gene is $L > 0$ G((([lim6n_lwt] + [lim6c_lwt]) > [thres_h]) && (similarAbsolute(([lim6n_rwt] + [lim6c_rwt]), 0, [thres_l]))) {[time] > 250}	Fig. 3A of ref. 43; Fig. 4A and 5B of ref. 33
Rule 19	In *die-1*(*ot26*) mutants, the expression pattern of *lim-6* gene is $0=0$ G (similarAbsolute(([lim6n_lot26] + [lim6c_lot26]), 0, [thres_l]) && similarAbsolute(([lim6n_rot26] + [lim6c_rot26]), 0, [thres_l])) {[time] > 250}	Fig. 3A of ref. 43; Fig. 5B of ref. 33
Rule 20	In *cog-1*(*sy607*) mutants, the expression pattern of *lim-6* gene is $L = R$ G (similarAbsolute(([lim6n_lsy607] + [lim6c_lsy607]), ([lim6n_rsy607] + [lim6c_rsy607]), [thres_l]) && !(similarAbsolute(([lim6n_rsy607] + [lim6c_rsy607]), 0, [thres_l]))) {[time] > 250}	Fig. 3A of ref. 43
Rule 21	In *die-1*(*ot26*);*cog-1*(*sy607*) double mutants, the expression pattern of *lim-6* gene is $0=0$ G (similarAbsolute(([lim6n_lot26sy607] + [lim6c_lot26sy607]), 0, [thres_l]) && similarAbsolute(([lim6n_rot26sy607] + [lim6c_rot26sy607]), 0, [thres_l])) {[time] > 250}	
Rule 22	In *lsy-6*(*ot71*) mutants, the expression pattern of *lim-6* gene is $0=0$ G (similarAbsolute(([lim6n_lot71] + [lim6c_lot71]), 0, [thres_l]) && similarAbsolute(([lim6n_rot71] + [lim6c_rot71]), 0, [thres_l])) {[time] > 250}	Fig. 3A of ref. 43; Fig. 5B of ref. 33
Rule 23	In the wild type, the expression pattern of *gcy-5* gene is $0 < R$ G (([gcy5_rwt] > [thres_h]) && similarAbsolute([gcy5_lwt], 0, [thres_l])) {[time] > 250}	Fig. 3B of ref. 43; Fig. 4B and D of ref. 33
Rule 24	In *die-1*(*ot26*) mutants, the expression pattern of *gcy-5* gene is $L = R$ G (similarAbsolute([gcy5_lot26], [gcy5_rot26], [thres_l]) && !(similarAbsolute([gcy5_rot26], 0, [thres_l]))) {[time] > 250}	Fig. 3B of ref. 43
Rule 25	In *lsy-6*(*ot71*) mutants, the expression pattern of *gcy-5* gene is $L = R$ G (similarAbsolute([gcy5_lot71], [gcy5_rot71], [thres_l]) && !(similarAbsolute([gcy5_rot71], 0, [thres_l]))) {[time] > 250}	
Rule 26	There exist no expressions of ASEL/ASER-specific reporter genes (*i.e.*, *gcy-5*, *flp-4*,*gcy-6*, *hen-1*) in ASEL neuron X (similarAbsolute([gcy5_lwt], 0, [thres_h]) && similarAbsolute([flp4_lwt], 0, [thres_h]) && similarAbsolute([gcy6_lwt], 0, [thres_h]) && similarAbsolute([hen1_lwt], 0, [thres_h])) {[time] = = 0}	ref. 32
Rule 27	In the wild type, the expression pattern of *gcy-6*/*gcy-7* gene is $L > 0$ G (([gcy6_lwt] > [thres_h]) && similarAbsolute([gcy6_rwt], 0, [thres_l])) {[time] > 250}	Fig. 4A of ref. 33
Rule 28	In *fozi-1*(*cc607*) mutants, the expression pattern of *gcy-6*/*gcy-7* gene adopts $L > R$ or $L = R$ G((([gcy6_lcc607] > ([gcy6_rcc607] + [thres_h])) && !(similarAbsolute([gcy6_rcc607], 0, [thres_l]))) ‖ (similarAbsolute([gcy6_lcc607], [gcy6_rcc607], [thres_l]) && !(similarAbsolute([gcy6_rcc607], 0, [thres_l])))) {[time] > 250}	
Rule 29	In the wild type, the expression pattern of *flp-4* gene adopts $0=0$ or $L > 0$ G ((similarAbsolute([flp4_lwt], 0, [thres_l]) && similarAbsolute([flp4_rwt], 0, [thres_l])) ‖ (([flp4_lwt] > [thres_h]) && similarAbsolute([flp4_rwt], 0, [thres_l]))) {[time] > 250}	
Rule 30	In *fozi-1*(*cc607*) mutants, the expression pattern of *flp-4* gene adopts $L > R$, $L = R$ or $L < R$ G ((([flp4_lcc607] > ([flp4_rcc607] + [thres_h])) && !(similarAbsolute([flp4_rcc607], 0, [thres_l]))) ‖ (similarAbsolute([flp4_lcc607], [flp4_rcc607], [thres_l]) && !(similarAbsolute([flp4_rcc607], 0, [thres_l]))) ‖ ((([flp4_lcc607] + [thres_h]) < [flp4_rcc607]) && !(similarAbsolute([flp4_lcc607], 0, [thres_l])))) {[time] > 250}	
Rule 31	In the wild type, the expression pattern of *hen-1* gene adopts $0=0$ or $0 < R$ G ((similarAbsolute([hen1_lwt], 0, [thres_l]) && similarAbsolute([hen1_rwt], 0, [thres_l])) ‖ (([hen1_rwt] > [thres_h]) && similarAbsolute([hen1_lwt], 0, [thres_l]))) {[time] > 250}	Fig. 4B of ref. 33

Table 4 (*continued*)

No.	Rules Translation	Biological evidences
Rule 32	In *fozi-1*(*cc607*) mutants, the expression pattern of *hen-1* gene adopts $0 = 0$ or $0 < R$ G ((similarAbsolute([hen1_lcc607], 0, [thres_l]) && similarAbsolute([hen1_rcc607], 0, [thres_l])) ‖ (([hen1_rcc607] > [thres_h]) && similarAbsolute([hen1_lcc607], 0, [thres_l]))) {[time] > 250}	
Rule 33	In *fozi-1*(*cc607*) mutants, the expression pattern of *gcy-5* gene adopts $0 < \mathrm{R}$ G (([gcy5_rcc607] > [thres_h]) && similarAbsolute([gcy5_lcc607], 0, [thres_l])) {[time] > 250}	
Rule 34	In *fozi-1*(*cc607*) mutants, the expression pattern of *lsy-6* gene adopts $0 = 0$, $L > 0$, $L > R$, $L = R$, $0 < R$ G ((similarAbsolute(([lsy6n_lcc607] + [lsy6c_lcc607]), 0, [thres_l]) && similarAbsolute(([lsy6n_rcc607] + [lsy6c_rcc607]), 0, [thres_l])) ‖ (((([lsy6n_lcc607] + [lsy6c_lcc607]) > [thres_h]) && similarAbsolute(([lsy6n_rcc607] + [lsy6c_rcc607]), 0, [thres_l])) ‖ (((([lsy6n_lcc607] + [lsy6c_lcc607]) > (([lsy6n_rcc607] + [lsy6c_rcc607]) + [thres_h])) && !(similarAbsolute(([lsy6n_rcc607] + [lsy6c_rcc607]), 0, [thres_l]))) ‖ ((similarAbsolute(([lsy6n_lcc607] + [lsy6c_lcc607]), ([lsy6n_rcc607] + [lsy6c_rcc607]), [thres_l]) && !(similarAbsolute(([lsy6n_rcc607] + [lsy6c_rcc607]), 0, [thres_l])))) ‖ (((([lsy6n_rcc607] + [lsy6c_rcc607]) > [thres_h]) && similarAbsolute(([lsy6n_lcc607] + [lsy6c_lcc607]), 0, [thres_l]))) {[time] > 250}	Fig. 5D of ref. 33
Rule 35	In the wild type, the expression pattern of *cog-1* gene adopts $0 = 0$, $L < R$ or $0 < R$ G ((similarAbsolute(([cog1n_lwt] + [cog1c_lwt]), 0, [thres_l]) && similarAbsolute(([cog1n_rwt] + [cog1c_rwt]), 0, [thres_l])) ‖ (((([cog1n_lwt] + [cog1c_lwt]) + [thres_h]) < ([cog1n_rwt] + [cog1c_rwt])) && !(similarAbsolute(([cog1n_lwt] + [cog1c_lwt]), 0, [thres_l]))) ‖ ((([cog1n_rwt] + [cog1c_rwt]) > [thres_h]) && similarAbsolute(([cog1n_lwt] + [cog1c_lwt]), 0, [thres_l]))) {[time] > 250}	
Rule 36	In *fozi-1*(*cc607*) mutants, the expression pattern of *cog-1* gene adopts $0 = 0$ or $0 < R$ G ((similarAbsolute(([cog1n_lcc607] + [cog1c_lcc607]), 0, [thres_l]) && similarAbsolute(([cog1n_rcc607] + [cog1c_rcc607]), 0, [thres_l])) ‖ ((([cog1n_rcc607] + [cog1c_rcc607]) > [thres_h]) && similarAbsolute(([cog1n_lcc607] + [cog1c_lcc607]), 0, [thres_l]))) {[time] > 250}	
Rule 37	In the wild type, the expression pattern of *fozi-1* gene adopts $0 = 0$, $L > R$, $L = R$ or $L < R$ G ((similarAbsolute([fozi1_lwt], 0, [thres_l]) && similarAbsolute([fozi1_rwt], 0, [thres_l])) ‖ ([fozi1_lwt] > ([fozi1_rwt] + [thres_h])) ‖ (similarAbsolute([fozi1_lwt], [fozi1_rwt], [thres_l]) && !(similarAbsolute([fozi1_lwt], 0, [thres_l]))) ‖ (([fozi1_lwt] + [thres_h]) < [fozi1_rwt])) {[time] > 250}	Fig. 5A of ref. 33
Rule 38	In *die-1*(*ot26*) mutants, the expression pattern of *fozi-1* gene adopts $L > R$, $L = R$ or $L < R$ G (([fozi1_lot26] > ([fozi1_rot26] + [thres_h])) ‖ (similarAbsolute([fozi1_lot26], [fozi1_rot26], [thres_l]) && !(similarAbsolute([fozi1_lot26], 0, [thres_l]))) ‖ (([fozi1_lot26] + [thres_h]) < [fozi1_rot26])) {[time] > 250}	
Rule 39	In *lsy-6*(*ot71*) mutants, the expression pattern of *fozi-1* gene adopts $L > R$, $L = R$ or $L < R$ G (([fozi1_lot71] > ([fozi1_rot71] + [thres_h])) ‖ (similarAbsolute([fozi1_lot71], [fozi1_rot71], [thres_l]) && !(similarAbsolute([fozi1_rot71], 0, [thres_l]))) ‖ (([fozi1_lot71] + [thres_h]) < [fozi1_rot71])) {[time] > 250}	
Rule 40	In *cog-1*(*sy607*) mutants, the expression pattern of *fozi-1* gene adopts $0 = 0$, $L > R$, $L = R$ or $L < R$ G ((similarAbsolute([fozi1_lsy607], 0, [thres_l]) && similarAbsolute([fozi1_rsy607], 0, [thres_l])) ‖ ([fozi1_lsy607] > ([fozi1_rsy607] + [thres_h])) ‖ (similarAbsolute([fozi1_lsy607], [fozi1_rsy607], [thres_l]) && !(similarAbsolute([fozi1_rsy607], 0, [thres_l]))) ‖ (([fozi1_lsy607] + [thres_h]) < [fozi1_rsy607])) {[time] > 250}	
Rule 41	In *lim-6*(*nr2073*) mutants, the expression pattern of *fozi-1* gene adopts $L = R$ or $L < R$ G ((similarAbsolute([fozi1_lnr2073], [fozi1_rnr2073], [thres_l]) && !(similarAbsolute([fozi1_lnr2073], 0, [thres_l]))) ‖ (([fozi1_lnr2073] + [thres_h]) < [fozi1_rnr2073])) {[time] > 250}	
Rule 42	In *fozi-1*(*ot61*);*lsy-6*(*ot71*) double mutants, the expression pattern of *lim-6* gene adopts $L = R$ G ((similarAbsolute(([lim6n_lot61ot71] + [lim6c_lot61ot71]), ([lim6n_rot61ot71] + [lim6c_rot61ot71]), [thres_l]) && !(similarAbsolute(([lim6n_rot61ot71] + [lim6c_rot61ot71]), 0, [thres_l])))) {[time] > 250}	Fig. 5B of ref. 33
Rule 43	In *die-1*(*ot26*);*fozi-1*(*ot61*) double mutants, the expression pattern of *lim-6* gene adopts $L = R$ G (similarAbsolute(([lim6n_lot26ot61] + [lim6c_lot26ot61]), ([lim6n_rot26ot61] + [lim6c_rot26ot61]), [thres_l]) && !(similarAbsolute(([lim6n_rot26ot61] + [lim6c_rot26ot61]), 0, [thres_l]))) {[time] > 250}	
Rule 44	In *fozi-1*(*ot61*) mutants, the expression pattern of *lim-6* gene adopts $L > R$ or $L = R$ G(((([lim6n_lcc607] + [lim6c_lcc607]) > (([lim6n_rcc607] + [lim6c_rcc607]) + [thres_h])) && !(similarAbsolute(([lim6n_rcc607] + [lim6c_rcc607]), 0, [thres_l]))) ‖ ((similarAbsolute(([lim6n_lcc607] + [lim6c_lcc607]), ([lim6n_rcc607] + [lim6c_rcc607]), [thres_l])) && !(similarAbsolute(([lim6n_rcc607] + [lim6c_rcc607]), 0, [thres_l])))) {[time] > 250}	
Rule 45	In *lim-6*(*nr2073*) mutants, the expression pattern of *gcy-5* gene adopts $L = R$, $L < R$ or $0 < R$ G ((similarAbsolute([gcy5_lnr2073], [gcy5_rnr2073], [thres_l]) && !(similarAbsolute([gcy5_rnr2073],0, [thres_l]))) ‖ ((([gcy5_lnr2073] + [thres_h]) < [gcy5_rnr2073]) && !(similarAbsolute([gcy5_lnr2073], 0 , [thres_l]))) ‖ (([gcy5_rnr2073] > [thres_h]) && similarAbsolute([gcy5_lnr2073], 0, [thres_l]))) {[time] > 250}	Fig. 4D of ref. 33



```
G ((([mir273n_lwt]+[mir273c_lwt])>(([mir273n_rwt]+[mir273c_rwt])+[thres_h]))
|| similarAbsolute(([mir273n_lwt]+[mir273c_lwt]),([mir273n_rwt]+ [mir273c_rwt]), [thres_l])
|| ((([mir273n_lwt]+[mir273c_lwt])+[thres_h])<([mir273n_rwt]+[mir273c_rwt]))){[time]>250}
```

Fig. 6 PLTL statement formulated from observed biological results. (a) The asymmetric expression of *mir-273*^{prom}*::gfp* in wild type (see Fig. 2A of ref. ref. 43), The National Academy of Sciences of the USA ©2005. (b) Biological result in a PLTL syntax.

Table 5 Summary of verifying three simulation models against 45 rules given in Table 4. SKIP: Skipped due to missing substances not found in the model; T: True; F: False

No.	**Model1** (Saito *et al.* 2005)	**Model2** (Extended **Model1**)	**Model3**
Rule 1	T	T	T
Rule 2	T	T	T
Rule 3	T	T	T
Rule 4	T	T	T
Rule 5	T	**F**	T
Rule 6	T	T	T
Rule 7	T	T	T
Rule 8	T	T	T
Rule 9	T	T	T
Rule 10	T	T	T
Rule 11	T	T	T
Rule 12	T	T	T
Rule 13	T	T	T
Rule 14	**F**	**F**	T
Rule 15	**F**	**F**	T
Rule 16	T	T	T
Rule 17	T	T	T
Rule 18	T	**F**	T
Rule 19	T	T	T
Rule 20	T	T	T
Rule 21	T	T	T
Rule 22	T	T	T
Rule 23	T	**F**	T
Rule 24	T	T	T
Rule 25	T	T	T
Rule 26	T	T	T
Rule 27	T	**F**	T
Rule 28	SKIP	**F**	T
Rule 29	T	**F**	T
Rule 30	SKIP	T	T
Rule 31	T	T	T
Rule 32	SKIP	T	T
Rule 33	SKIP	**F**	T
Rule 34	SKIP	T	T
Rule 35	T	T	T
Rule 36	SKIP	T	T
Rule 37	SKIP	T	T
Rule 38	SKIP	**F**	T
Rule 39	SKIP	T	T
Rule 40	SKIP	T	T
Rule 41	SKIP	T	T
Rule 42	SKIP	**F**	T
Rule 43	SKIP	**F**	T
Rule 44	SKIP	T	T
Rule 45	T	T	T
TRUE	29	33	45
FALSE	2	12	0
SKIP	14	0	0

stochastic noise with respect to 21 estimated parameters. We observe that 13 parameters (*i.e.*, *e29*, *e30*, *e31*, *e33*, *e34*, *e35*, *e37*, *e38*, *e39*, *e40*, *e41*, *e481*, *e482*) can conform to 45 biological criteria for the entire simulation runs even against the highest standard deviation value;

(ii) For Model3_14, the entity *e484* is highly unstable but still possesses a high score of satisfied rule numbers. It is likely that such entity storing the reaction rate of *fozi-1* is important to the regulation of ASE fate decision.

Finally, we count the number of satisfied rules of five standard deviations and calculate their average values for 57 models with respect to 21 parameters. From the resulting plot shown in Fig. 8, we conclude that:

(i) the average values of *e7* and *e9* are significantly unstable against the noise strengths. These two parameters are the threshold values of inhibitory regulations induced from two miRNAs (*lsy-6* miRNA and *mir-273* miRNA). It can be considered that the regulations caused by the miRNAs are essential to the regulation of forming alternative cell fates;

(ii) the input (*e7*) and the output regulation (*e40*) of the double-negative feedback loop constituted by *lsy-6*, *cog-1*, *die-1* and *mir-273* are easy to be in an unstable condition by the perturbations;

(iii) the initial value of *lsy-6* (*e13*) is a sensitive factor in spite of its low concentration compared to [lsy2c];

(iv) the inhibitory regulations from [die1c] to [fozi1] (*e29*) and from [fozi1] to [lim6c2] (*e481*) also have surprising fluctuations, and it generates the following view that these regulations are maybe closely linked to the ASEL/ASER development.

Discussions

Data assimilation

When constructing a simulation model of biological pathways, we usually tune the unknown kinetic parameters manually so as to fit the observed data. Obviously, the hand tuning approach has limitations in terms of the size of the network and labor time. To extend the capability of the simulation model, Nagasaki *et al.* have proposed the so-called *data assimilation* (DA) approach.[45] Their DA framework enables users to handle both the model construction and the parameter tuning within statistical inferences, and establishes a link between the HFPNe simulation model and observed data, *e.g.*, microarray gene expression data[45] or time series proteomic data.[46,47] Although this approach has succeeded in constructing simulation models of circadian rhythm in *mouse*[45] and epidermal growth factor receptor signal transduction pathway model (EGFR model),[46,47] it would be impossible for statistical inferences because of the difficulty of providing time-course data compulsively involving successive time points

Table 6 Summary of range values for parameter estimation. Please refer to the legend of Fig. 5 for the notation of the substance in the brackets. The entities used for parameter estimation are indicated in blue in Fig. 4 for quick reference

Entity in model	Name	Type	Range values
e7	Threshold value of the inhibition from [lsy6c] to [cog1c]	Threshold value	0–0.25
e11	Threshold value of the inhibition from [die1c] to [gcy5]	Threshold value	0–0.35
e13	Initial value of [lsy6n]	Initial value	0–0.02
e28	Threshold value of the inhibition from [lim6c] to [gcy5]	Threshold value	0–0.7
e29	Threshold value of the inhibition from [die1c] to [fozi1]	Threshold value	0–0.26
e30	Threshold value of the inhibition from [fozi1] to [lim6c1]	Threshold value	0–0.55
e31	Threshold value of the inhibition from [fozi1] to [gcy6]	Threshold value	0–0.55
e32	Threshold value of the inhibition from [die1c]to [hen1]	Threshold value	0–0.2
e33	Threshold value of the association from [lsy2n] to [lsy6n]	Threshold value	0–0.1
e34	Threshold value of the association from [die1n] to [lsy6n]	Threshold value	0–0.1
e35	Threshold value of the association from [lim6n] to [lsy6n]	Threshold value	0–0.1
e36	Threshold value of the association from [lim6n] to [lim6mRNA]	Threshold value	0–0.1
e37	Threshold value of the association from [die1c] to [lim6c]	Threshold value	0–0.1
e38	Threshold value of the association from [die1c] to[flp4]	Threshold value	0–0.1
e39	Threshold value of the association from [die1c] to [gcy6]	Threshold value	0–0.1
e40	Threshold value of the association from [cog1n] to [mir273n]	Threshold value	0–0.1
e41	Threshold value of the association from [cog1n] to [cog1mRNA]	Threshold value	0-0.1
e9	Threshold value of the inhibition from [mir273c] to [die1c]	Threshold value	0–0.25
e481	Threshold value of the inhibition from [fozi1] to [lim6c2]	Threshold value	0–0.55
e482	Threshold value of the association from [lim6c] to [flp4]	Threshold value	0–0.1
e477	High threshold value	Threshold value	0–0.1
e478	Low threshold value	Threshold value	0–0.5
e484	Transcription speed of *fozi-1*	Reaction rate	0–0.25

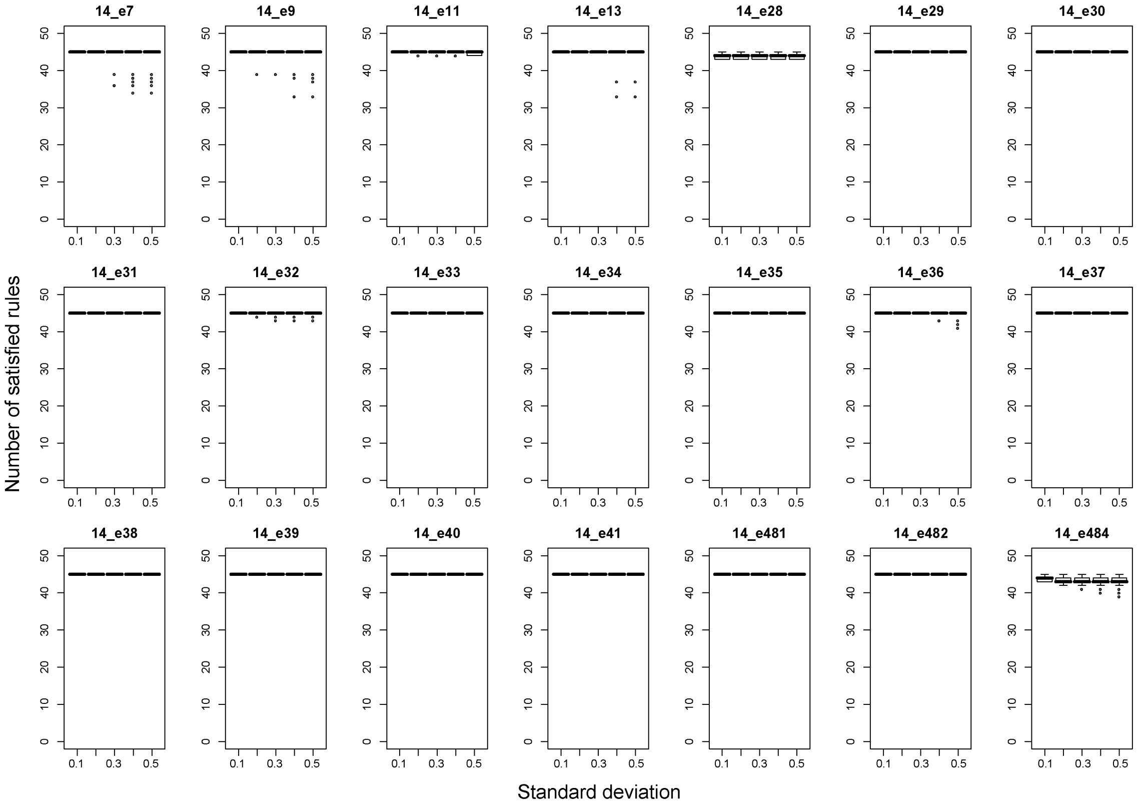


Fig. 7 The plots regarding simulation results of Model3_14 under five standard deviations of 21 parameters. *x*-Axis denotes the strength of noise from 0.1 to 0.5; *y*-axis is the number of satisfied rules.

(at least 10–20 time points) by biological experiments. That is, for a small time-course data set, including for example 2 or 3 time points, DA does not work effectively. The response to this difficulty is the use of model checking, as the model checking approach is not restricted to the completeness and continuousness of the observed time-course data or quantified dynamic profile due to the expressive power of temporal logic for formal verification.

Related researches

We have shown here the combination of our online model checker MIRACH and parameter estimation method on the

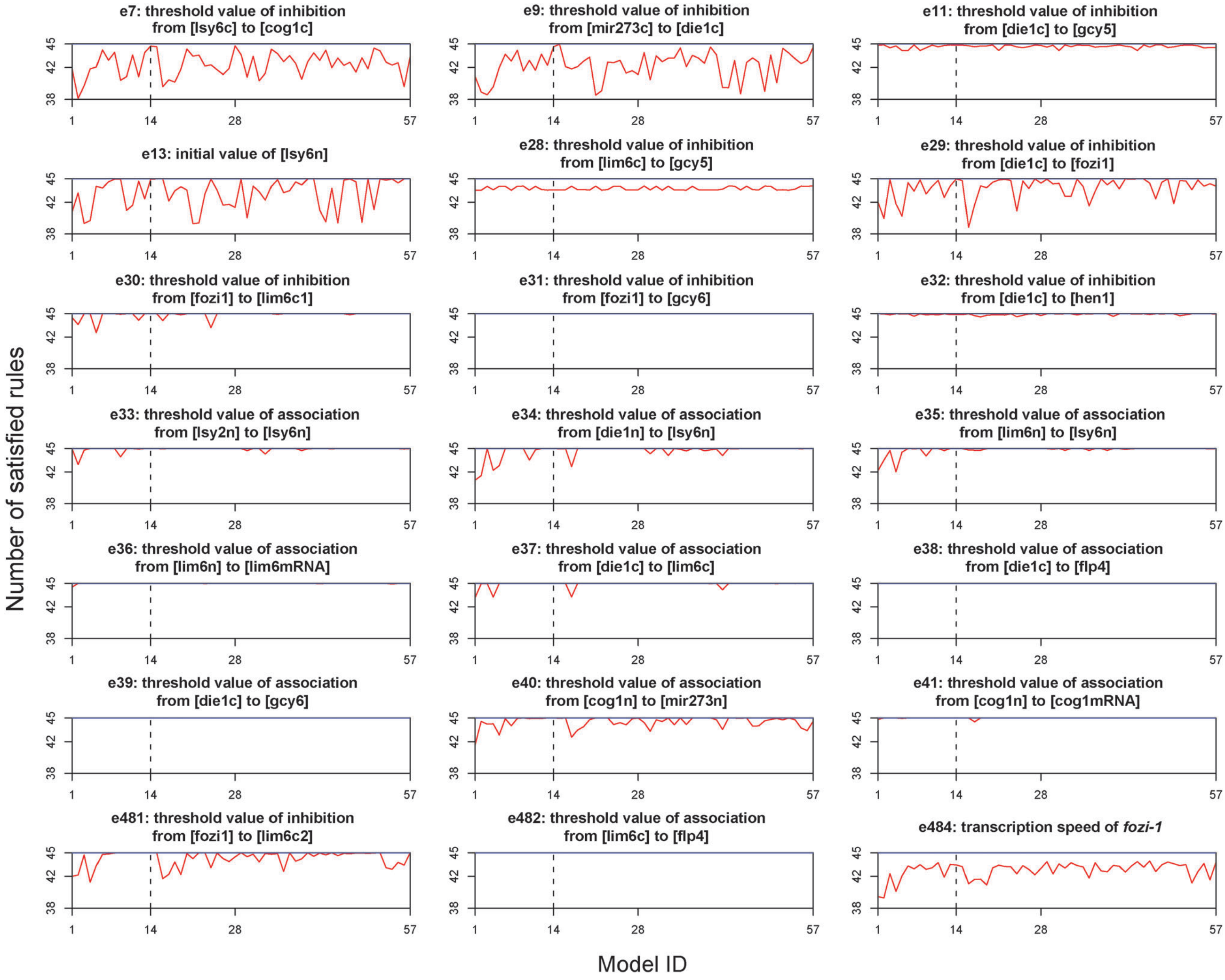


Fig. 8 The average values of 57 stochastic models for each 21 estimated parameter. Note that each average value (*i.e.* each point in the chart) is calculated by adding together the satisfied rule number of five standard deviations for each parameter and each model.

class of HFPNe. It affords us with plausible and robust simulation models to aid us in understanding system mechanisms. Donaldson and Gilbert have developed a computational system MC2(GA),[40] which incorporates a model checker called MC2(PLTLc)[39] supporting another temporal logic called probabilistic LTL with numerical constrains (PLTLc) coupled with a genetic algorithm for parameter estimation. MC2(PLTLc) is based on the offline model checking approach and did not use any statistics to compute returned results' confidence whereas it is realized in MIRACH although it is not demonstrated in this case study. As previously mentioned, offline model checking would be a waste of CPU resource if the decision of validity or rejection for the simulation run could be determined early in its execution. In ref. 40, when offline model checking is coupled with the genetic algorithm to do parameter estimation, the total number of simulation runs would increase significantly because of increasing parameters to record and longer running time resulting from every model at each generation/iteration. For more detailed comparisons, please see Koh *et al.*[41]

On the other hand, Heiner *et al.*[31] have applied simulative model checking of PLTL to a Petri net class, *i.e.*, extended stochastic Petri nets, but parameter estimation has not been taken into account.

Conclusions

This paper describes a systematic framework to automatically estimate kinetic parameters of a given model or a model starting from scratch by applying an online model checking technique supported by temporal logic PLTL and its extension to a well-founded Petri net class (*i.e.*, HFPNe), and a simple yet sufficient parameter estimation method. Its main contributions lie, firstly, in building a quantitative ASE fate model in *C. elegans* with 3327 components emulating nine genetic conditions. This model extends a previous model of Saito *et al.*[32] by taking into account a newly identified transcription factor *fozi-1*. Secondly, we have extracted 45 biological properties in total underlying dynamic behaviours about ASE fate determination from observed known data which are later translated into PLTL formulas for the formal verification. In this study, the utilization of a probability operator has not been given in demonstrating the applicability to an ASE fate model. Nevertheless, our online model checker MIRACH supports PLTL and its extension in statistic settings. Thirdly, we have conducted a large number of simulations (20 million-run) to estimate 23 kinetic parameters contributing to the regulation of forming alternative cell fates, and obtained 57 parameter sets whose models can conform to the entire 45 specifications for further simulation analysis. Finally, we have evaluated the correctness and robustness of these 57 models by adding noise of different magnitudes into the models. One simulation model is concluded to be the most reasonable and robust owing to the high stability under the disturbance. We also have discussed the results and summarized several plausible explanations.

As far as we know, this is the first attempt that connects a model checking approach with a parameter estimation technique to a high-level Petri net class for biological pathways. We focus here on the establishment of the systematic framework combining these two methods and briefly adopted a uniform distribution for parameter estimation, which allows other more accurate estimation methods to be used as well.

Acknowledgements

This research was supported by the postdoctoral fellowship program of Japan Society for the Promotion of Science (JSPS). And we are deeply thankful for the computational resources provided by the Super Computer System, Human Genome Center, Institute of Medical Science, University of Tokyo, Japan.

Funding: Singapore National Research Foundation grant NRF-G-CRP-2997-04-082(d) (C. H. K., in parts); National University of Singapore NGS scholarship (to C. H. K., in parts).

Notes and references

1 H. Yan, B. Zhang, S. Li and Q. Zhao, A formal model for analyzing drug combination effects and its application in TNF-alpha-induced NFkappaB pathway, *BMC Syst. Biol.*, 2010, **4**, 50.

2 M. Antoniotti, A. Policriti, N. Ugel and B. Mishra, Model building and model checking for biochemical processes, *Cell Biochem. Biophys.*, 2003, **38**(3), 271–286.

3 A. Regev, W. Silverman and E. Shapiro, Representation and simulation of biochemical processes using the pi-calculus process algebra, *Pac. Symp. Biocomput*, 2001, **6**, 459–470.

4 S. C. Peng, D. S. Wong, K. C. Tung, Y. Y. Chen, C. C. Chao, C. H. Peng, Y. J. Chuang and C. Y. Tang, Computational modeling with forward and reverse engineering links signaling network and genomic regulatory responses: NF-kappaB signaling-induced gene expression responses in inflammation, *BMC Bioinf.*, 2010, **11**, 308.

5 M. Chen and R. Hofestädt, Quantitative Petri net model of gene regulated metabolic networks in the cell, *In Silico Biol.*, 2003, **3**(3), 347–365.

6 I. Koch, B. H. Junker and M. Heiner, Application of Petri net theory for modelling and validation of the sucrose breakdown pathway in the potato tuber, *Bioinformatics*, 2005, **21**(7), 1219–1226.

7 J. Wu and E. Voit, Hybrid modeling in biochemical systems theory by means of functional Petri nets, *J. Bioinf. Comput. Biol.*, 2009, **1**, 107–134.

8 S. Hardy and P. N. Robillard, Petri net-based method for the analysis of the dynamics of signal propagation in signaling pathways, *Bioinformatics*, 2008, **24**(2), 209–217.

9 G. Koh, H. F. Teong, M. V. Clement, D. Hsu and P. S. Thiagarajan, A decompositional approach to parameter estimation in pathway modeling; a case study of the Akt and MAPK pathways and their crosstalk, *Bioinformatics*, 2006, **22**(14), 271–280.

10 D. Ruths, M. Muller, J. T. Tseng, L. Nakhleh and P. T. Ram, The signaling Petri net-based simulator: a non-parametric strategy for characterizing the dynamics of cell-specific signaling networks, *PLoS Comput. Biol.*, 2008, **4**(2), e1000005.

11 A. Sackmann, D. Formanowicz, P. Formanowicz and J. Blazewicz, New insights into the human body iron metabolism analyzed by a Petri net based approach, *BioSystems*, 2009, **96**(1), 104–113.

12 S. Troncale, J. P. Comet and G. Bernot, Validation of Biological Models with Temporal Logic and Timed Hybrid Petri Nets, *Conf. Proc. IEEE Eng. Med. Biol. Soc.*, 2007, 4603–4608.

13 J. Wu, Z. Qi and E. Voit, Impact of delays and noise on dopamine signal transduction, *In Silico Biol.*, 2010, **10**, 0005.

14 C. Li, M. Nagasaki, A. Saito and S. Miyano, Time-dependent structural transformation analysis to high-level Petri net model with active state transition diagram, *BMC Syst. Biol.*, 2010, **4**, 39.

15 L. J. Steggles, R. Banks, O. Shaw and A. Wipat, Qualitatively modelling and analysing genetic regulatory networks: a Petri net approach, *Bioinformatics*, 2007, **23**(3), 336–343.

16 C. Li, M. Nagasaki, K. Ueno and S. Miyano, Simulation-based model checking approach to cell fate specification during *Caenorhabditis elegans* vulval development by hybrid functional Petri net with extension, *BMC Syst. Biol.*, 2009, **3**, 42.

17 C. Chaouiya, Petri net modelling of biological networks, *Briefings Bioinf.*, 2007, **8**(4), 210–219.

18 I. Koch and M. Heiner, *Petri nets*, in *Analysis of Biological Networks*, ed. B. H. Junker and F. Schreiber, A Wiley Interscience Publication, 2008, pp. 139–180.

19 E. M. Clarke, O. Grumberg and D. A. Peled, *Model Checking*, The MIT Press, 1999.

20 B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, P. Schnoebelen and P. McKenzie, *Systems and Software Verification: Model-Checking Techniques and Tools*, Springer, 2001.

21 E. M. Clarke and E. A. Emerson, *Design and synthesis of synchronization skeletons using branching time temporal logic*, in *Logic of Programs*, 1981, pp. 52–71.

22 J. P. Queille and J. Sifakis, Specification and verification of concurrent systems in CESAR, *Lect. Notes Comput. Sci.*, 1982, **137**, 337–351.

23 N. Chabrier and F. Fages, *Symbolic model checking of biochemical networks*, in *Proceedings of CMSB 2003 (Computational Methods in Systems Biology), Italy*, Springer Berlin/Heidelberg, Italy, 2003, pp. 149–162.

24 G. Batt, D. Ropers, H. de Jong, J. Geiselmann, R. Mateescu, M. Page and D. Schneider, Validation of qualitative models of genetic regulatory networks by model checking: analysis of the nutritional stress response in *Escherichia coli*, *Bioinformatics*, 2005, **21**(Suppl 1), i19–i28.

25 M. Calder, V. Vyshemirsky, D. Gilbert and R. Orton, *Analysis of signalling pathways using the prism model checker*, in *Proceedings of CMSB 2005 (Computational Methods in Systems Biology)*, ed. G. Plotkin, University of Edinburgh, 2005, pp. 179–190.

26 J. Fisher, N. Piterman, A. Hajnal and T. A. Henzinger, Predictive modeling of signaling crosstalk during *C. elegans* vulval development, *PLoS Comput. Biol.*, 2007, **3**(5), e92.

27 J. Heath, M. Kwiatkowska, G. Norman, D. Parker and O. Tymchyshyn, Probabilistic model checking of complex biological pathways, *Theor. Comput. Sci.*, 2008, **391**(3), 239–257.

28 M. Z. Kwiatkowska, G. Norman and D. Parker, Using probabilistic model checking in systems biology, *SIGMETRICS Performance Evaluation Review*, 2008, **35**(4), 14–21.

29 P. T. Monteiro, D. Ropers, R. Mateescu, A. T. Freitas and H. de Jong, Temporal logic patterns for querying dynamic models of cellular interaction networks, *Bioinformatics*, 2008, **24**(16), i227–i233.

30 N. Bonzanni, E. Krepska, K. A. Feenstra, W. Fokkink, T. Kielmann, H. Bal and J. Heringa, Executing multicellular differentiation: quantitative predictive modelling of *C. elegans* vulval development, *Bioinformatics*, 2009, **25**(16), 2049–2056.

31 M. Heiner, S. Lehrack, D. Gilbert and W. Marwan, *Extended stochastic Petri nets for model-based design of wetlab experiments*,

in *Trans. on Comput. Syst. Biol. XI, LNBI*, ed. C. Priami *et al.*, 2009, vol. 5750, pp. 138–163.

32 A. Saito, M. Nagasaki, A. Doi, K. Ueno and S. Miyano, Cell fate simulation model of gustatory neurons with MicroRNAs double-negative feedback loop by hybrid functional Petri net with extension, *Genome Inf.*, 2006, **17**(1), 100–111.

33 R. J. Jr Johnston, J. W. Copeland, M. Fasnacht, J. F. Etchberger, J. Liu, B. Honig and O. Hobert, An unusual Zn-finger/FH2 domain protein controls a left/right asymmetric neuronal fate decision in C. elegans, *Development (Cambridge, UK)*, 2006, **133**(17), 3317–3328.

34 M. Nagasaki, A. Saito, E. Jeong, C. Li, K. Kojima, E. Ikeda and S. Miyano, Cell Illustrator 4.0: A computational platform for systems biology, *In Silico Biol.*, 2010, **10**, 0002.

35 M. Nagasaki, A. Doi, H. Matsuno and S. Miyano, A versatile Petri net based architecture for modeling and simulation of complex biological processes, *Genome Inf.*, 2004, **15**(1), 180–197.

36 A. H. Hawari and Z. A. Mohamed-Hussein, Simulation of a Petri net-based model of the terpenoid biosynthesis pathway, *BMC Bioinf.*, 2010, **11**(83).

37 R. Alur, T. A. Henzinger, F. Y. C. Mang, S. Qadeer, S. K. Rajamani and S. Tasiran, MOCHA: modularity in model checking, *Lect. Notes Comput. Sci.*, 1998, **1427**, 521–525.

38 E. M. Clarke, J. R. Faeder, C. J. Langmead, L. A. Harris, S. K. Jha and A. Legay, *Statistical model checking in BioLab: applications to the automated analysis of T-cell receptor signaling pathway*, *Proc. CSMB 2008*, pp. 231–250.

39 R. Donaldson and D. Gilbert, A Monte Carlo model checker for probabilistic LTL with numerical constraints, Technical report, University of Glasgow, Department of Computing Science, 2008.

40 R. Donaldson and D. Gilbert, *A model checking approach to the parameter estimation of biochemical pathways*, *Proc. CSMB 2008*, pp. 269–287.

41 C. H. Koh, M. Nagasaki, A. Saito, C. Li, L. Wong and S. Miyano, MIRACH: Efficient Model Checker for Quantitative Biological Pathway Models, *Bioinformatics*, 2011, **27**, 734.

42 C. Jard and T. Jeron, On-line model-checking for finite linear temporal logic specifications, *Lect. Notes Comput. Sci.*, 1990, **407**, 189–196.

43 R. J. Jr Johnston, S. Chang, J. F. Etchberger, C. O. Ortiz and O. Hobert, MicroRNAs acting in a double-negative feedback loop to control a neuronal cell fate decision, *Proc. Natl. Acad. Sci. U. S. A.*, 2005, **102**(35), 12449–12454.

44 O. Hobert, Architecture of a microRNA-controlled gene regulatory network that diversifies neuronal cell fates, *Cold Spring Harbor Symp. Quant. Biol.*, 2006, **71**, 181–188.

45 M. Nagasaki, R. Yamaguchi, R. Yoshida, S. Imoto, A. Doi, Y. Tamada, H. Matsuno, S. Miyano and T. Higuchi, Genomic data assimilation for estimating hybrid functional Petri net from time-course gene expression data, *Genome Inf.*, 2006, **17**(1), 46–61.

46 S. Tasaki, M. Nagasaki, M. Oyama, H. Hata, K. Ueno, R. Yoshida, T. Higuchi, S. Sugano and S. Miyano, Modeling and estimation of dynamic EGFR pathway by data assimilation approach using time series proteomic data, *Genome Inf.*, 2006, **17**(2), 226–238.

47 S. Tasaki, M. Nagasaki, H. Kozuka-Hata, K. Semba, N. Gotoh, S. Hattori, J. Inoue, T. Yamamoto, S. Miyano, S. Sugano and M. Oyama, Phosphoproteomics-based modeling defines the regulatory mechanism underlying aberrant EGFR signaling, *PLoS One*, 2010, **5**(11), e13926.