Briefings in Bioinformatics, 22(1), 2021, 438–450

doi: 10.1093/bib/bbz114
Advance Access Publication Date: 15 December 2019
Problem solving protocol

Hybrid modelling of biological systems using fuzzy continuous Petri nets

Fei Liu, Wujie Sun, Monika Heiner and David Gilbert

Corresponding author: Fei Liu, School of Software Engineering, South China University of Technology, Guangzhou 510006, P.R. China. Tel.:+86-20-39380068; Fax: +86-20-39380068; E-mail: feiliu@scut.edu.cn

Abstract

Integrated modelling of biological systems is challenged by composing components with sufficient kinetic data and components with insufficient kinetic data or components built only using experts' experience and knowledge. Fuzzy continuous Petri nets (FCPNs) combine continuous Petri nets with fuzzy inference systems, and thus offer an hybrid uncertain/certain approach to integrated modelling of such biological systems with uncertainties. In this paper, we give a formal definition and a corresponding simulation algorithm of FCPNs, and briefly introduce the FCPN tool that we have developed for implementing FCPNs. We then present a methodology and workflow utilizing FCPNs to achieve hybrid (uncertain/certain) modelling of biological systems illustrated with a case study of the Mercaptopurine metabolic pathway. We hope this research will promote the wider application of FCPNs and address the uncertain/certain integrated modelling challenge in the systems biology area.

Key words: systems biology; integrated modelling; fuzzy continuous Petri nets; uncertainties; hybrid simulation.

Introduction

Modelling and simulation play an essential role in the study of systems biology by constructing mathematical or computational models of biological systems, which help biologists to better understand and predict system behaviour [1, 2]. Recently, high-throughput experimental technologies for biological systems have rapidly developed and we have obtained an increasingly deeper understanding of biological mechanisms. Thus, more and more complicated biological models have been constructed to study the behaviour of systems from a holistic point of view. In this case, integrated modelling of biological systems is becoming crucial [3], which incorporates into one model different types of biological networks, built by more than one modelling method. Furthermore, whole-cell modelling [4, 5] has been proposed, which attempts to incorporate all the essential genes and processes and their interactions of a cell into one model.

Integrated modelling of biological systems faces many challenges, e.g. different components may have distinct structures and characteristics and available kinetic data and knowledge may also vary quite a lot [6]. By incorporating these heterogeneous components to achieve an integrated model, hybrid methods are necessary [7]. Hybrid methods [8] do by definition integrate more than one modelling formalism into one model,

Fei Liu is an associate Professor at the School of Software Engineering, South China University of Technology. His research interests are modelling and simulation, Petri nets and systems biology.
Wujie Sun is an undergraduate student at the School of Software Engineering, South China University of Technology. His research interests are modelling and simulation and machine learning.
Monika Heiner is a Professor at the Department of Computer Science, Brandenburg University of Technology Cottbus-Senftenberg. Her research interests include modelling and analysis of technical as well as biochemical networks using qualitative and quantitative Petri nets, model checking and simulation techniques.
David Gilbert is a Professor at the Department of Computer Science, Brunel University London. His research interests include Bioinformatics, Systems Biology, Synthetic Biology, multiscale modelling, model checking and computational methods for the design of biological systems.
Submitted: 19 May 2019; **Received (in revised form):** 30 July 2019

thus permitting the adoption of an appropriate formalism for each specific component according to its characteristics.

So far, many modelling formalisms have been proposed to address different issues encountered in systems biology, which can be categorized in several ways [6, 7, 9]: qualitative or quantitative, discrete or continuous, deterministic or stochastic, certain or uncertain. For example, ordinary differential equations (ODEs) and continuous Petri nets (CPNs) [10] can be classified as quantitative, continuous, deterministic and certain. Standard Petri nets [11] are qualitative and discrete. Fuzzy rules are qualitative, discrete and uncertain.

Combinations of two or more of these formalisms result in hybrid methods, which aim to address more complicated (biological) modelling issues. For example, one popular combination is to integrate stochastic and deterministic formalisms, in which Gillespie's stochastic simulation algorithm [12] and numerical ODE solvers are adopted to solve the constructed hybrid models [13]. Such a combination is by its very nature still one of the quantitative approaches. Hybrid functional Petri nets [14] are another hybrid method that integrates discrete Petri nets and ODEs. These hybrid methods are essential to achieve integrated modelling of biological systems, but they are by far not sufficient.

Currently, modelling of biological systems is inevitably affected by uncertainties due to the lack of kinetic data or insufficient understanding of the inherent mechanisms of biological systems [6]. As a result, integrated modelling is challenged by composing components with sufficient kinetic data and components with insufficient kinetic data or components built only using experts' experience and knowledge. Fuzzy continuous Petri nets (FCPNs) are a promising method to address this challenge.

FCPNs [9, 15, 16] combine CPNs [11] with fuzzy logic [17] and offer a hybrid uncertain/certain approach to modelling and analysing a complex biological system, where some components are built as a set of ODEs, if kinetic data are sufficiently well known, and the others as a set of uncertain fuzzy inference systems (FISs) by resorting to the experience and knowledge of biologists, if kinetic data are insufficiently known or completely unavailable. The concepts and analysis methods of FCPNs have been discussed in [15, 16] and further compared in [9]. FCPNs have been used for modelling the green fluorescent protein expression in a cell-free in vitro transcription/translation system [15] and a hypothetical repressilator with three genes [16]. This previous work well illustrates the power of FCPNs. In [9], we classified the wide variety of fuzzy Petri nets into three categories: basic fuzzy Petri nets (full structural uncertainty), fuzzy quantitative Petri nets (partial structural uncertainty) and Petri nets with fuzzy kinetic parameters (only parametric uncertainty). FCPNs as discussed in this paper fall into the second category of fuzzy Petri nets. That is, an FCPN model is divided into two parts, the certain ODE part and the uncertain FIS part.

FCPNs are very helpful for constructing and analysing biological models by integrating both quantitative kinetic data and qualitative expert knowledge. However, FCPNs have been used so far for very few applications in this area. This could be because there are neither a formal definition nor simulation algorithm or easy-to-use tool of FCPNs. Currently, Matlab is usually used for building FCPN models with its fuzzy toolbox and ODE solvers. This, however, has many drawbacks, e.g. it requires writing code and has no simple user interface. These drawbacks hinder the wider use of FCPNs by biologists.

To address these issues, we formally defined FCPNs and designed a related simulation algorithm. We also developed a graphical tool, called FCPN tool, for modelling and simulating of biological networks with FCPNs. In this paper, we will focus on presenting a methodology and workflow about how to utilize FCPNs to achieve hybrid (uncertain/certain) modelling of biological systems illustrated with a case study of the mercaptopurine metabolic pathway. We hope this research will promote the wider application of FCPNs and to some extent address the hybrid uncertain/certain modelling challenge in the systems biology area.

Materials and methods

Fuzzy inference systems

Fuzzy logic [17, 18] was proposed to deal with vagueness and imprecise information and simulate human reasoning, which has been proved to be a helpful tool to address the uncertainty modelling issue due to the lack of data or insufficient understanding of systems by taking advantage of the experience of experts.

The basic concept of fuzzy logic is the fuzzy set, which allows an element to partially belong to a set, i.e. each element in a set is given a membership degree between 0 and 1. Formally, a fuzzy set A is defined by its membership function μ_A over a universal set U, i.e. $\mu_A : U \to [0, 1]$. In contrast, in a classical crisp set, an element is either a member of the set or not, i.e. the membership degree of an element is either 0 or 1.

Commonly used membership functions include triangular, trapezoidal and Gaussian membership functions. The former two are piecewise linear, continuous functions, while the latter one is a smooth differentiable function. A linguistic value or term is a fuzzy set defined on a universe, e.g. the set of real numbers, and is usually used for defining fuzzy rules. For example, to describe the concentration of a species, we may define three linguistic terms, Low, Medium and High, each taking a triangular membership function.

Fuzzy set operations are used for generating new fuzzy sets. Let A and B be two fuzzy sets defined on a common universe U. The fuzzy union of A and B, denoted by $A \cup B$, is usually defined as $\mu_{A\cup B}(x) = \mu_A(x) \vee \mu_B(x)$, and the fuzzy intersection of A and B, denoted by $A \cap B$, is defined as $\mu_{A\cap B}(x) = \mu_A(x) \wedge \mu_B(x)$, where $x \in U$.

Fuzzy sets and fuzzy set operations can be considered as the subjects and verbs of fuzzy logic, respectively. Further, fuzzy IF-THEN rules are used to project input variables onto output space. A single fuzzy IF-THEN rule takes the following form:

$$\text{IF } x \text{ is } A \text{ and } y \text{ is } B, \text{ THEN } z \text{ is } C. \tag{1}$$

The IF part is called the antecedent or premise, where x and y are input variables. The THEN part is called the consequent, where z is the output variable. A, B and C are linguistic terms defined over their respective universe sets.

A FIS is used to achieve the mapping from a given input to an output using fuzzy logic. There are two major types of FISs, the Mamdani FIS [19] and the Takagi–Sugeno (T-S) FIS [20]. These two FISs are similar in many aspects, e.g. fuzzifying inputs and applying fuzzy operators during reasoning. The main difference is that the T-S output membership functions are either linear or constant, while the Mamdani output membership functions are fuzzy.

An FIS usually consists of four components (see Figure 1): fuzzifier, fuzzy rule base, inference engine and defuzzifier (does not apply to T-S FIS).

The fuzzifier is responsible for comparing the crisp values of input variables with the membership functions on the

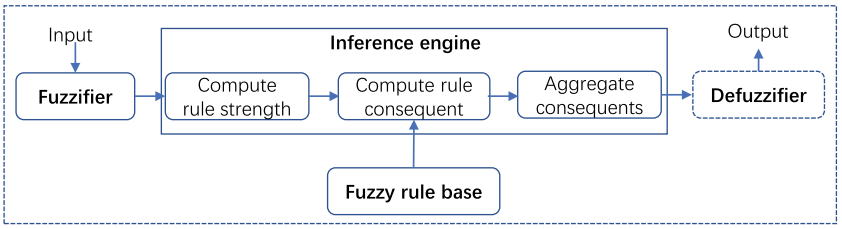

Figure 1. The main components of a fuzzy inference system. The fuzzifier is used to fuzzify crisp input values to fuzzy values. The fuzzy rule base stores all the fuzzy rules that model a system. An inference engine performs the following three steps: compute the firing strength of each rule, compute the consequent of a rule by combining the rule strength and the output membership function and aggregate the consequents of all rules to obtain the output value. Finally, a crisp output is obtained via the defuzzifier; this only applies to the Mamdani inference. Except the defuzzifier, all other components apply to both types of FISs. Figure 2 gives an FIS example to illustrate this reasoning process.

antecedent part to obtain the membership values of each linguistic term.

The fuzzy rule base contains a set of fuzzy IF-THEN rules that model a system. For a Mamadani FIS, the fuzzy rules take the form given in Eq. 1. But for a T-S FIS, the fuzzy rules take the following form:

$$\text{IF } x \text{ is } A \text{ and } y \text{ is } B, \text{ THEN } z = f(x, y), \tag{2}$$

where $f(x, y)$ is usually a polynomial function.

The inference engine combines the membership values on the antecedent part to get the firing strength (or weight) of each rule, denoted by α, usually by means of minimum or multiplication operations, e.g. $\alpha = \mu_A \wedge \mu_B$ for the rule given in Eq. 1. It then generates the qualified consequent of each rule, and finally combines the fuzzy sets that represent the output of each rule into a single fuzzy set for the Mamdani inference, or combines the crisp output of each rule into a single output value for the T-S inference.

For the Mamdani inference, there is another component, the defuzzifier, which converts the aggregated output fuzzy set to a crisp output value using such methods as centroid of area. See Figure 2 for an FIS example that illustrates the reasoning process sketched in Figure 1.

The individual advantages of the Mamdani inference and T-S inference are as follows. The Mamdani FIS is intuitive and well-suited to human input and reasoning, which is appropriate to help biologists to construct fuzzy models by using their experience when there are only a few kinetic data and not much understanding about the biological mechanisms. In contrast, the T-S FIS is computationally more efficient, and can easily take advantage of adaptive techniques to learn its parameters when there are sufficient kinetic data available.

In our approach, we consider a special class of FISs, where each FIS is defined as a multi-input and single-output FIS: $y = f(\mathbf{x})$, where y is the output and $\mathbf{x} = (x_1, x_2, \ldots)$ are inputs.

Learning of fuzzy rules

Fuzzy modelling based on FISs is an effective method for representing uncertain systems; however, this method has one main drawback—it is hard to identify fuzzy rules and tune the membership functions of the FISs (especially for T-S FISs). Thus, neural networks (NNs) have been combined with FISs to form fuzzy neural networks (FNNs) by utilizing the learning capability of NNs. With FNNs, we can acquire the fuzzy rules and tune membership functions simultaneously by learning from data. So far, many FNNs have been proposed (see a recent review in [21]), e.g. the adaptive neuro-fuzzy inference system [22], which is widely used to train T-S fuzzy rules.

In the following, we briefly describe how to construct a FNN to train Mamdani fuzzy rules [23] by considering the following four rules.

Rule 1: IF x_1 is A_1 and x_2 is B_1, THEN y is C_1.
Rule 2: IF x_1 is A_1 and x_2 is B_2, THEN y is C_2.
Rule 3: IF x_1 is A_2 and x_2 is B_1, THEN y is C_3.
Rule 4: IF x_1 is A_2 and x_2 is B_2, THEN y is C_4.

The corresponding FNN is shown in Figure 3, comprising five layers, namely the input layer, membership function layer, rule layer, normalization layer and output layer.

(1) In the input layer, each node corresponds to an input variable, such as x_1 or x_2.

(2) In the membership function layer, each node corresponds to a linguistic term, such as A_1 or B_1. This layer calculates the membership degree μ of an input value, e.g. x_1, belonging to its corresponding fuzzy set, e.g. A_1. Take the Gaussian function as an example. The degree μ can be calculated using Eq. 3.

$$\mu_{ij} = \exp\left(-\frac{(x_i - c_{ij})^2}{2\sigma_{ij}^2}\right) \tag{3}$$

where $i = 1, \ldots, m$ and $j = 1, \ldots, n_i$. m is the number of input variables and n_i is the number of linguistic terms of the i^{th} input variable x_i. c_{ij} and σ_{ij} are the center and variance of the corresponding membership function, respectively. In this example, m and n_i are equal to 2.

(3) In the rule layer, each node corresponds to a rule. The firing strength α can be calculated using Eq. 4.

$$\alpha_k = \mu_{1p_1} \wedge \cdots \wedge \mu_{ip_i} \tag{4}$$

where $k = 1, \ldots, q$, $i = 1, \ldots, m$ and $p_i \in [1, \ldots, n_i]$. q is the number of rules. In this example, q is equal to 4. α_k can be obtained by calculating the minimum value from μ_{1p_1} to μ_{ip_i}.

(4) In the normalization layer, we normalize the firing strengths of fuzzy rules using Eq. 5.

$$\bar{\alpha}_k = \frac{\alpha_k}{\sum_{k=1}^{q} \alpha_k} \tag{5}$$

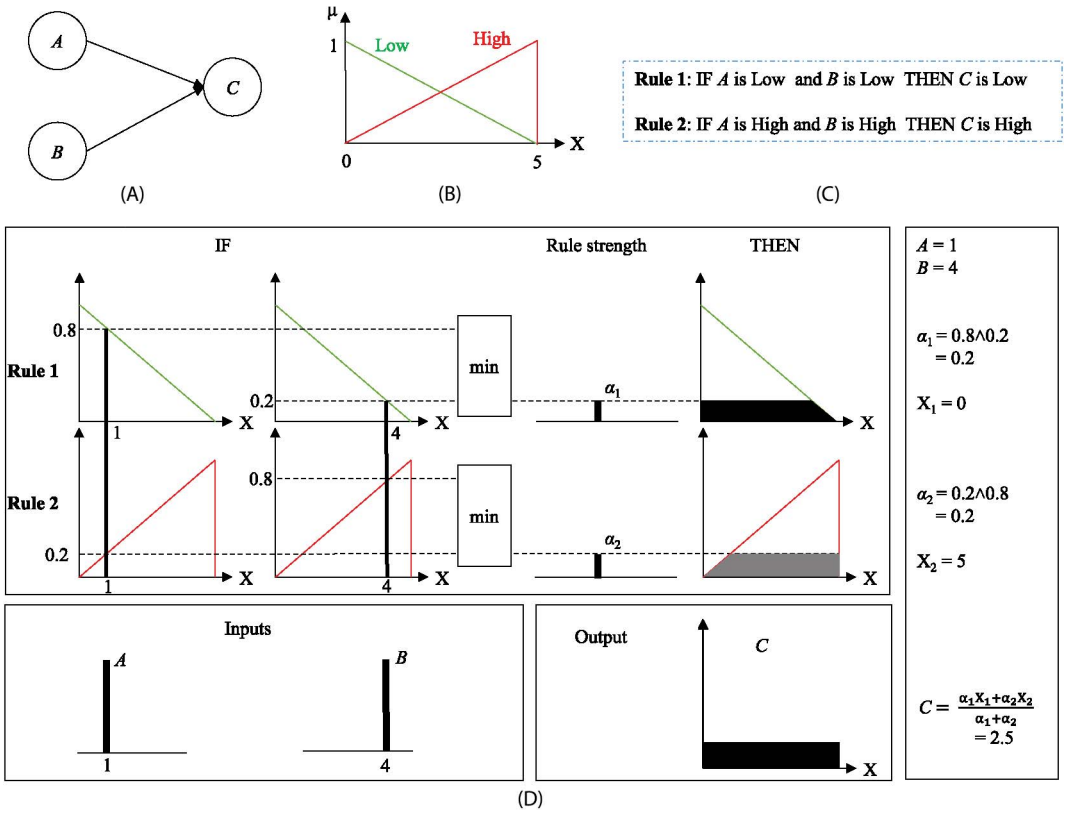

Figure 2. An FIS example. **(A)** A simple model consisting of three proteins, A, B and C. **(B)** Two linguistic terms, Low and High, each taking a triangular membership function, which are used to describe the concentration levels of these three proteins. **(C)** Two fuzzy rules that describe how the concentration of C changes according to the concentrations of A and B. **(D)** A Mamdani inference process by applying the FIS given in Figure 1.

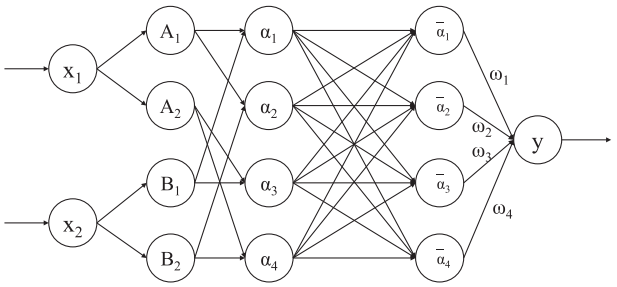

Figure 3. An FNN example that represents the four fuzzy rules given above.

(5) In the output layer, we perform the defuzzification operation using Eq. 6.

$$y = \sum_{k=1}^{q} \omega_k \bar{\alpha}_k \tag{6}$$

Since we use the centroid defuzzification method, ω_k is the linguistic value of the k^{th} rule's output whose corresponding degree of membership is equal to 1. Its initial value is obtained by a random initialization.

The FNN for Mamdani fuzzy rules uses the back-propagation algorithm and gradient descent. The error can be calculated using Eq. 7.

$$Error = \frac{1}{2}\left(y_{actual} - y_{predict}\right)^2 \tag{7}$$

When we construct an FNN, we first train it with data, and thus obtain parameters for all membership functions.

Continuous Petri nets

Petri nets [24, 25] are weighted, directed, bipartite multigraphs. A Petri net consists of two kinds of nodes, places and transitions, and arcs that connect these two kinds of nodes. In the biological scenario, places usually represent chemical species or other specific compounds, and transitions represent their reactions or interactions. The tokens residing on places, given as non-negative integers, represent the number of molecules or concentration levels of species.

Petri nets have been extended in many ways to adapt them to different scenarios. One extension is CPNs [26], in which tokens are allowed to be non-negative real values that represent, e.g. the concentration of species or compounds. Transitions now fire continuously, if at all. Thus, both places and transitions are not discrete any more, but continuous.

A CPN model represents graphically a set of ODEs and thus the underlying semantics of a CPN is a system of ODEs, where each equation describes the continuous token change for a given place, i.e. the continuous increase by its pretransitions' flow and the continuous decrease by its posttransitions' flow. An equation for a place p has the following form:

$$\frac{dm(p)}{d\tau} = \sum_{t \in {}^\bullet p} f(t,p)v(t) - \sum_{t \in p^\bullet} f(p,t)v(t),$$

where ${}^\bullet p$ and $p^\bullet$ denote the pretransitions and posttransitions of a place p, respectively, $f(t,p)$ and $f(p,t)$ the multiplicity of the arc (t,p) and (p,t), respectively, and $v(t)$ the rate of the transition t.

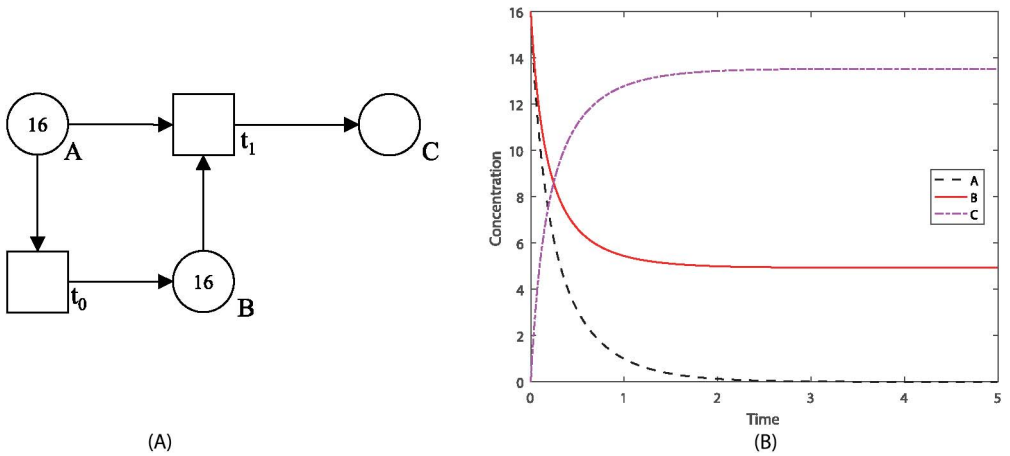

Figure 4. (**A**) A CPN model of a feed-forward loop [29], which consists of three genes. A transcription factor *A* regulates a co-factor *B*, both of which jointly regulate a target gene *C*. (**B**) Simulation plot of the model. The values of the kinetic parameters for t_0 and t_1 are arbitrarily set to $k_0 = 0.5$ and $k_1 = 0.3$, respectively.

During simulation, the CPN models are automatically transformed in the background into ODEs and then solved using standard ODE solvers (see [27] for a detailed description); therefore, there is no difference in terms of the computation speed between CPNs and ODEs. Petri nets including CPNs have been used for constructing large models such as whole genome scale metabolic models (GEMs) [28].

For example, Figure 4A gives a CPN model of a feed-forward loop. This model generates the following set of ODEs, when applying the mass action semantics for the transition rates. Note that we arbitrarily set the kinetic parameters (here 0.3, 0.5) for this illustrative CPN example.

$$\begin{cases} dA/dt = -(0.5 * A) - (0.3 * A * B), \\ dB/dt = (0.5 * A) - (0.3 * A * B), \\ dC/dt = 0.3 * A * B. \end{cases}$$

When running numerical simulation for this CPN model, we obtain a simulation result as illustrated in Figure 4B.

Fuzzy continuous Petri nets

The concept of combining ODEs (or CPNs) with fuzzy rules was proposed in [15, 16]. However, these previous articles neither provide a formal definition nor a simulation algorithm. To clarify the concept and precise semantics of FCPNs, we give in the following the formal definition of an FCPN, which builds on the formal definition for CPNs given in [26], however extended by FISs.

An FCPN is a six-tuple $N =< P, T, F, f, v, m_0 >$, where:

- P is a finite, non-empty set of continuous places.
- T is a finite, non-empty set of continuous transitions.
- P and T are disjunctive.
- $F \subseteq (P \times T) \cup (T \times P)$ is a finite set of directed arcs.
- $f : F \to \{\mathbb{R}^+ \cup \{FIS\}\}$ is a function that assigns a non-negative real value or an FIS to each arc $a \in F$. Each FIS is defined as a multi-input and single-output FIS: $y = f(\mathbf{x})$, where y is the output and $\mathbf{x} = (x_1, x_2, \cdots)$ are inputs, with $y, x_i \in P$. Each FIS is either a Mamdani or T-S FIS.
- $v : T \to H$ is a function that assigns a firing rate function h_t to each transition $t \in T$, whereby $H := \bigcup_{t \in T}\{h_t | h_t : \mathbb{R}^{+|{}^\bullet t|} \to \mathbb{R}\}$ is the set of all firing rate functions, and $v(t) = h_t$ for all transitions $t \in T$. ${}^\bullet t$ denotes all the preplaces of t. $\mathbb{R}$ and $\mathbb{R}^+$ denote the set of real numbers and the set of all non-negative real numbers, respectively.
- $m_0 : P \to \mathbb{R}^+$ gives the initial marking.

The most important difference between our FCPN model and those given in [15, 16] is that both only allow Mamdani FISs, but we incorporate both Mamdani and T-S FISs. Thus, when we model a biological system, we can choose either Mamdani or T-S FIS for a specific component, depending on its characteristics and the availability of kinetic data. Similar to [15], we allocate FISs to arcs, while in [16] an FIS is associated with a transition. A detailed comparison about where to hold an FIS can be found in [9].

In our definition, we achieve the fuzzy modelling by associating an FIS with an arc and we only allow an FIS to be a multi-input and single-output system. Assume we want to assign an FIS to an arc a, and the transition that a connects is denoted by t whose preplaces and postplaces are denoted by ${}^\bullet t$ and $t^\bullet$, respectively. If a connects a place $p \in {}^\bullet t$, the input of the FIS associated with a is the concentration of only p and the output is the concentration change of p, as this FIS describes the concentration decrease of p due to only p. If a connects a place $p \in t^\bullet$, the inputs of the FIS associated with a are the concentrations of all ${}^\bullet t$ and the output is the concentration change of p, as this FIS describes the concentration increase of p due to all preplaces of t.

That is, an FIS models how the concentrations of the inputs produce the change (increment or decrement) of the output. With FISs, we can incorporate expert knowledge and experience into a quantitative model, offering a semi-quantitative method for constructing larger and more complete models even when some components lack kinetic data.

For example, Figure 5A gives an FCPN model by adapting the CPN model given in Figure 4A. The arc from transition t_0 to place *B* has a T-S FIS, called FIS(A;B), which describes the change of species *B* in terms of species *A*. FIS(A;B) is defined by the membership functions given in Figure 5B and the fuzzy rules given in Figure 5C. The arc from transition t_1 to place *C* has a Mamdani FIS, called FIS(A,B;C), which describes the change of species *C* in terms of species *A* and *B*. FIS(A,B;C) is defined by the membership functions given in Figure 5B and the fuzzy rules given in Figure 5D. Apart from these two FISs, the FCPN is described by a set of ODEs.

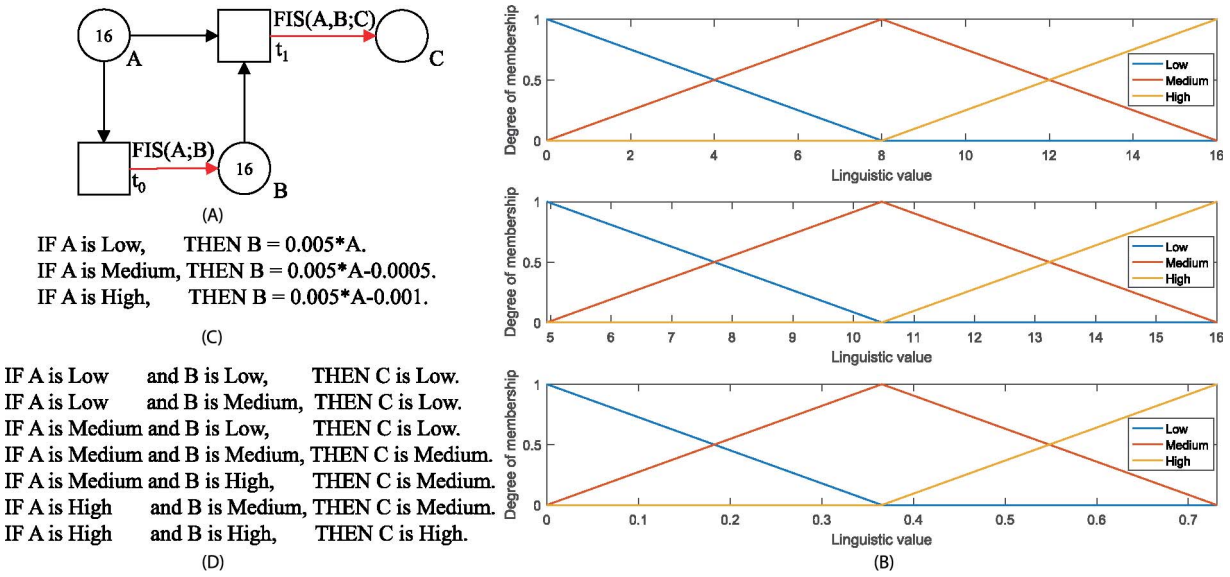

Figure 5. (**A**) An FCPN model of a feed-forward loop [29]. (**B**) Membership functions of the fuzzy sets describing the concentrations of A and B, and the concentration change of C (from top to bottom). (**C**) A set of T-S fuzzy rules for defining FIS(A;B). (**D**) A set of Mamdani fuzzy rules for defining FIS(A,B;C).

If we remove $\{FIS\}$ from $f : F \to \{\mathbb{R}^+ \cup \{FIS\}\}$, then an FCPN degrades to a CPN. The semantics of an FCPN is defined by a set of ODEs, a set of FISs and the interplay between ODEs and FISs; so we may have to resort to simulation to check its dynamic behaviour. When numerically solving the set of ODEs, at each simulation step we have to reason each FIS to update the state of the model. By repeating this process, we finally obtain the dynamic behaviour of the model. This will be discussed in detail in the next section.

Hybrid simulation algorithm

The simulation algorithm for our FCPNs is given in Algorithm 1, which works as follows.

Algorithm 1. Hybrid simulation algorithm

1: Set the initial simulation time to t_0;
2: Assume the initial marking of the model is C_0;
3: Set the fixed time step to $\delta(t)$ and the number of simulation steps to N;
4: Let $t = t_0$;
5: **for** $n = 1$ to N **do**
6: $t = t + \delta(t)$; //Advance simulation time
7: Let $C_n = C_{n-1}$;
8: **for** each place p **do**
9: Numerically solve the ODE associated with p for C_{n-1};
10: Obtain the change of p, denoted by $\Delta^1(p)$;
11: Update $C_n(p) = C_n(p) + \Delta^1(p)$;
12: **for** each FIS associated with p **do**
13: Reason the FIS for C_{n-1};
14: Obtain the change of p, denoted by $\Delta^2(p)$;
15: Update $C_n(p) = C_n(p) + \Delta^2(p)$;
16: **end for**
17: **end for**
18: Let $n = n + 1$;
19: **end for**

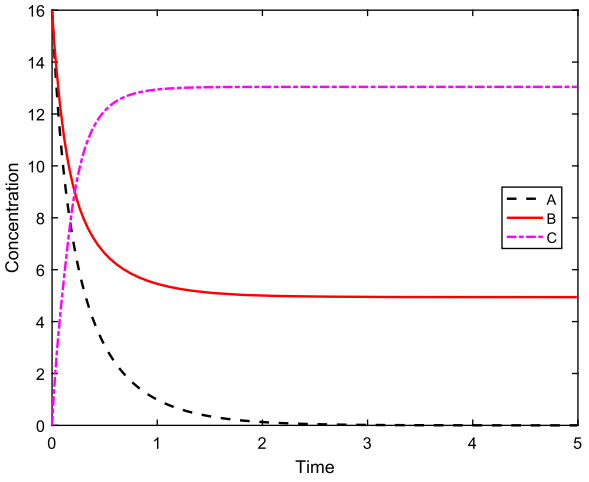

Figure 6. A hybrid simulation plot of the FCPN model given in Figure 5A. The mean squared errors of A, B and C between Figure 6 and Figure 4B are computed as 2.6788E-06, 1.24899E-04 and 2.52172E-01, respectively, i.e. the hybrid traces closely approximate the corresponding ODE traces.

At each time step n, for each place we perform the following two computational steps in parallel, although we give these two steps in the pseudocode sequentially. One is to numerically solve the ODE associated with each place p using any numerical solvers with fixed steps, e.g. the Runge–Kutta method, then obtain the change (increment or decrement) of p, denoted by $\Delta^1(p)$, and update the concentration of p in terms of $C_n(p) = C_n(p) + \Delta^1(p)$ (compare lines 9–11).

The other step is to perform reasoning of each FIS associated with a place p according to the following steps (compare lines 12–16, and see Figure 1 for an example):

i. Fuzzify the given input places of the FIS from crisp to fuzzy values.
ii. Calculate the fuzzy concentration change of the output place p via the inference engine by applying fuzzy rules.
iii. Defuzzify the fuzzy value of p to a crisp one.

iv. Update the concentration of the place p according to the concentration change resulting from the FIS reasoning, i.e. $C_n(p) = C_n(p) + \Delta^2(p)$.

The simulation algorithm precisely describes the operational semantics of FCPNs. For example, if we simulate the FCPN model given in Figure 5A, we obtain a hybrid simulation plot, illustrated in Figure 6, which is almost consistent with the ODE plot given in Figure 4B, but obtained with less knowledge resp. weaker assumptions.

FCPN tool

To facilitate the use of FCPNs, we developed a graphical tool, called FCPN tool, for modelling and simulating biological networks with FCPNs. FCPN tool provides functions for constructing FCPN models, for simulating them, and for plotting and exporting simulation results. The tool allows to graphically create FCPN models by simply dragging nodes (places or transitions) to the palette and then connect appropriate nodes with arcs. Each place holds a real number as token value, describing the concentration of a species. In the tool, we achieve the fuzzy modelling by associating an FIS with an arc. When opening the dialogue to edit the properties of an arc and then clicking the FIS button, we can enter the fuzzy setting dialogue, where we provide easy-to-use interfaces to specify the inputs and output of an FIS, to define their membership functions and to write fuzzy rules. Moreover, the tool offers two kinds of membership functions, triangular and Gauss, and two popular inference methods, Mamdani and T-S inference systems to define an FIS, which can be solely or together applied in one model. Besides, we can export the structure of an FIS (including specified inputs and outputs, types of membership functions, etc.) from the FCPN tool to a Matlab file, and then achieve the learning of fuzzy rules in the Matlab environment.

When an FCPN model is constructed, the simulation can be triggered by clicking the simulation button in the main window and then the simulation dialogue is opened. When simulation is finished, we can view the plot or data of the chosen places. Moreover, the tool allows to export plots to popular figure formats such as eps, pdf and png, to export data to a csv file, or to directly print plots.

The tool is written in the programming language C++ with QT 5.6.1. Currently, we offer windows, Linux and Mac OS X versions. The FCPN tool, user manual and a couple of examples can be found at https://github.com/liufei2016/FCPN.

In addition, we implemented the FNNs for both Mamdani and T-S fuzzy rules using Matlab, which achieve learning fuzzy rules and parameters. We also used Charlie [30] for structural analysis of FCPN models and the MC2 tool [31] for simulative model checking of simulation traces. Using a combination of these tools achieves an efficient approach for modelling and analysing of FCPN models.

Results

Workflow for using FCPNs

In this section, we will give a workflow of using FCPNs for modelling and analysing biological systems, which is shown in Figure 7. In the following, we first introduce a running example for illustrating our workflow and then in detail describe each element of the workflow.

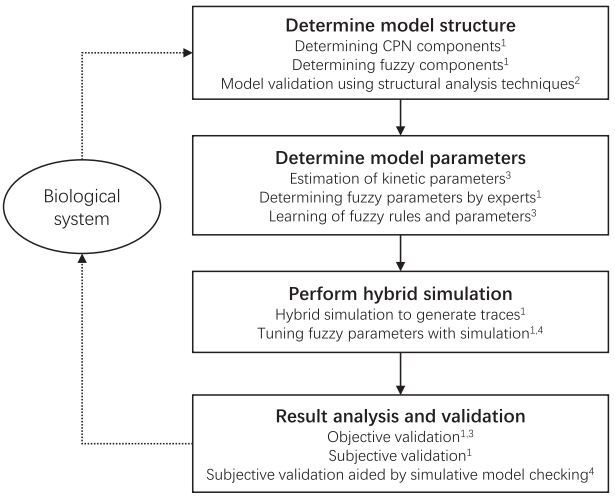

Figure 7. Workflow of modelling and analysing of biological systems using FCPNs. First, the model structure is determined by drawing an FCPN model and validated using Petri net analysis techniques. Second, the parameter estimation methods for ODE models are adopted to obtain parameter values if kinetic data are available; otherwise, experts are asked to specify fuzzy parameters directly, or FNNs are constructed to learn fuzzy rules and parameters. After these two steps, the model is constructed. Third, simulation is performed in different settings and results can be obtained with appropriate plots. Fourth, the model is analysed and validated against the real biological system. When the model passes validation, it can be used for behaviour prediction. To support the workflow, we adopt the following tools: [1]FCPN tool, [2]Charlie, [3]Matlab and [4]MC2.

A running example

The thiopurine antimetabolites mercaptopurine (MP) is one of the important chemotherapy drugs for treating acute lymphocytic leukemia (ALL). It undergoes complicated metabolic intracellular transformations that result in the production of thionucleotides and active metabolites. In this paper, we consider the mercaptopurine metabolic pathway given in [32], which works as follows [32, 33]. Intracellular mercaptopurine (MPex) goes into a cell after cellular uptake, and then the mercaptopurine inside a cell (MPin) is converted into TIMP (6-Thioinosine-5'-monophosphate) by hypoxanthine-guanine phosphoribosyl transferase. After that, TIMP is converted into TXMP (6-Thioxanthosine-5'-monophosphate), then finally to TGMP (6-Thioguanosine monophosphate). The transformation of TIMP to TITP (6-Thioinosine-5'-triphosphate) is another considered pathway. The original modelling purpose in [32] was to reveal the principal dynamics of the 6-MP metabolic transformations during the treatment of ALL and further to explore how ATP regulates the production of TITP and affects the transition of TXMP to TGMP.

However, we are not following this objective. Rather we will convert the ODE model given in [32] into an FCPN model, and illustrate the validity of our FCPN approach by comparing the result of the FCPN model with that of the original ODE model in [32]. In the following, we will use this pathway as our running example to illustrate our approach step by step.

Determine the model structure

The first step to construct a model is to determine its structure by carefully analysing the biological system to be modelled and collecting key concepts (e.g. species and reactions) from literature or by asking relevant experts. We then turn these species

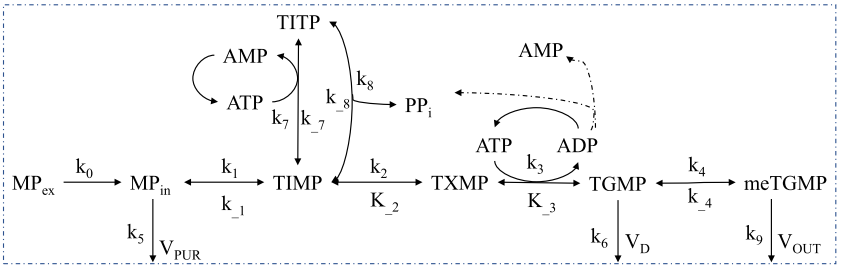

Figure 8. Mercaptopurine metabolic pathway. MP_{ex}—mercaptopurine outside of cell; MP_{in}—mercaptopurine entering a cell after cellular uptake; TIMP—6-Thioinosine-5'-monophosphate; TITP—6-Thioinosine-5'-triphosphate; TXMP—6-Thioxanthine-5'-monophosphate; TGMP—6-Thioguanosine-5'-monophosphate; meTGMP—Methylthioguanosine monophosphate; ATP, ADP and AMP—adenosine tri-, di- and monophosphates; PPi—pyrophosphate; V_D, V_{PUR} and V_{OUT}—fluxes involving the incorporation to DNA and RNA of cells, the inhibition of purine biosynthesis *de novo* and the outflux to the environment, respectively; k_i—kinetic constants of reactions. The dotted lines connecting ADP with AMP and PPi mean that ADP is converted to AMP and PPi after several intermediate steps.

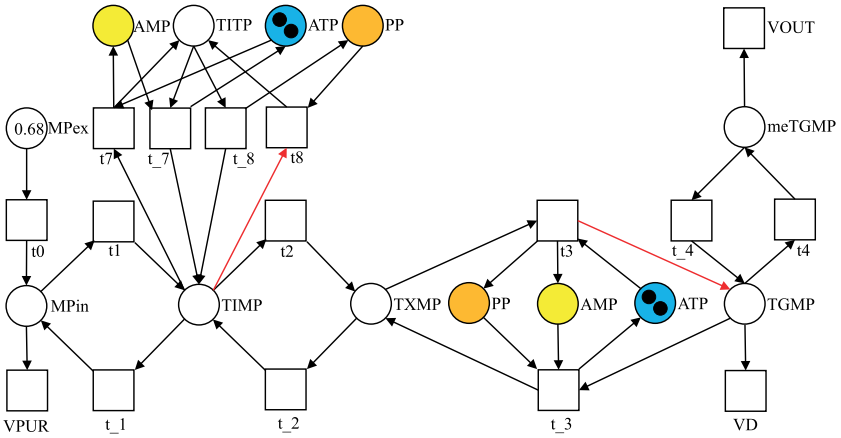

Figure 9. A Petri net model of the Mercaptopurine metabolic pathway, which describes the structure of the model only. To demonstrate the use of FCPNs, we set the arcs from TIMP to t8 and from t3 to TGMP (highlighted in red) to hold an FIS, Note that the three pairs of places with the same name (also given in the same colour) are logical nodes, which means they are actually the same node with two graphical representations.

and interactions into a Petri net model by representing each species as a place and each reaction as a transition and a couple of directed arcs according to the flow of the reaction. We further determine the model structure from the following two points of view.

Determine CPN components. For those components for which we have sufficient kinetic data and already understand their mechanisms, we try to obtain their chemical reactions or ODE representations and then model them as CPNs. Related aspects have been thoroughly discussed in [27, 34], e.g. how to derive ODEs from chemical reactions with mass-action or Michaelis–Menten kinetics, how to generate CPN models from ODEs, or vice versa.

Following the approaches given in [27, 34], we determine the model structure according to the scheme given in Figure 8 and the ODEs given in [32]; the resulting Petri net model is illustrated in Figure 9.

Determine fuzzy components. On the other hand, if we do not have a clear understanding of the biological mechanisms or sufficient kinetic data, we may model their dynamics with FISs. Another important scenario for using FISs is to connect different components described by ODEs with FISs, thus offering a possibility to produce larger models from isolated quantitative components. To create a fuzzy component, we may adopt the following simple procedure (see Figure 4 for an example): first determining the inputs and output of the component, then specifying membership functions of the inputs and output, and finally writing fuzzy (Mamdani or T-S) rules.

To demonstrate the use of FISs, we arbitrarily chose the arc from TIMP to t8 and the arc from t3 to TGMP to hold an FIS, respectively. Apart from these two arcs, the rest of the model is considered as an ODE/CPN model.

Model validation using structural analysis techniques. An important merit of FCPNs is to combine isolated qualitative and quantitative components of a biological system to form a larger model in order to gain deeper insights into the system behaviour. For such kind of hybrid models, it is essential but hard to verify them. As a special kind of Petri nets, FCPNs can easily take advantage of structural analysis techniques belonging to the standard body of Petri net theory, such as place/transition invariant or siphon/trap analysis; see [11] for details.

For this purpose, we export the model in Figure 9 as constructed with FCPN tool to a file of the ANDL format (our defined data exchange format), and feed it into Charlie [30], a Petri net analysis tool. We obtain the following analysis results.

The net is fully connected without any isolated components and structurally bounded (due to the source place MPex). Although it enjoys six minimal transition invariants (T-invariants for short) and two place invariants (P-invariants for short), it is neither covered with T-invariants nor with P-invariants. The six T-invariants exactly correspond to the six reversible reactions and the two P-invariants, (AMP,ATP) and (ATP,TITP,PP), reflect mass conservation.

However, if we add an input transition (named ti) as pre-transition to the place MPex, which means to add a continuous supply of mercaptopurine, the net gets three more non-trivial

T-invariants (i.e. not corresponding to reversible reactions), (ti,t0,VPUR), (ti,t0,t1,t2,t3,t_7,t8,VD) and (ti,t0,t1,t2,t3,t4,t_7,t8, VOUT), and is now covered with T-invariants. A T-invariant describes how often the transitions of the invariant have to fire to return to a given state. Moreover, T-invariants may be read to reflect the steady state behaviour in a metabolic network. These three non-trivial T-invariants exactly describe three steady state behaviours caused by the input transition ti and each of the three output transitions, VPUR, VOUT and VD. With structural analysis, we can easily find out and verify each basic behaviour (reflected by each invariant) existing in a model, which can be used for checking larger models by dividing them into basic components.

Besides, we observe that the Siphon-Trap Property (STP) holds in the net: the two P-invariants are the only minimal siphons in the model, which are by definition at the same time traps, and both are marked. The STP ensures that there are no dead states in the dynamic behaviour reachable.

After these structural analysis steps, we consider the mercaptopurine metabolic pathway model as validated. However, taking into account the usually finite supply of medicine (here mercaptopurine), we have to remove the newly added input transition ti for the further quantitative analysis.

Determine the model parameters

It is obvious that an FCPN model is divided into two parts: the CPN part and the fuzzy part. These two parts require different methods for determining their parameters.

Estimation of kinetic parameters. For the CPN part, after determining the model structure and automatically deriving the corresponding set of ODEs, we usually obtain the parameters from literature or estimated from a data fit by using well-established parameter estimation methods, see, e.g. [35, 36]. Parameter estimation or data fitting typically starts with a guess about parameter value ranges and then tunes those values to minimize the discrepancy between the model outputs and measured observations by help of a selected metric. There are many parameter estimation methods available to achieve this goal, which are usually categorized into two groups, the global group such as simulated annealing and evolutionary algorithms [37], and the local group such as gradient-based methods for least-squares [36]. We can choose the most appropriate methods according to the model structure and the available data.

In this paper, we focus on illustrating the application of FCPNs, so we do not perform parameter estimation for the mercaptopurine metabolic pathway model, but directly adopt the kinetic parameters given in [32]. Moreover, we first construct a pure CPN model based on [32] in order to compare the hybrid FCPN model to be constructed and its corresponding CPN model.

Determining fuzzy parameters by experts. For the fuzzy part, we may adopt different strategies to determine the parameters related to the membership functions defined for linguistic terms. If kinetic data is unavailable or insufficient, we may have to ask experts (biologists) to determine fuzzy parameters by hand with their experience and knowledge. On the contrary, if kinetic data is sufficient, we may automatically learn the parameters from the data. This section will describe the former situation.

(1) If kinetic data is unavailable or insufficient, we may ask experienced biologists to directly use simple (triangular) membership functions, which then allow us to perform simulations to see if the outputs of the model are consistent with the observations, or what the biologists expect to obtain. In this case, Mamdani FISs are usually preferred. The model in Figure 5 illustrates this strategy, where we directly specify the fuzzy parameters of triangular membership functions and fuzzy rules according to our experience, which can be imitated by anyone.

(2) If there are some (limited) measurements at specific time points available, which thus can generate concentrations of species, we can utilize this limited information to refine the fuzzy parameters of the membership functions and obtain more accurate fuzzy rules. To achieve this, we may adopt the following steps. Assume that we want to construct a fuzzy model whose input variables are x_1 and x_2, and the output variable is y, and we obtain measurements at some time points.

i. Compute the concentration change Δy of y at every time point relative to its previous time point.
ii. Estimate the concentration range of each input (x_1 and x_2) and output variable (Δy), perform fuzzy partitioning of the concentration range of each variable and specify a membership function (e.g. Low, Medium and High) for each partition.
iii. Construct a fuzzy rule at each time point in terms of the fuzzy partitioning obtained in the previous step. For example, at some time point, we may have such a rule: IF x_1 is High and x_2 is Medium, THEN Δy is High. Thus we can obtain a fuzzy model with limited measurements. See the supplementary information file for a detailed example.

Learning of fuzzy rules and parameters. If kinetic data are sufficient, but the biological mechanisms are not well understood, we may use FNNs to learn the fuzzy rules to obtain those fuzzy parameters, which have been discussed in detail in Section 2. In this case, either Mamdani or T-S fuzzy rules can be used. We may adopt the following procedure to achieve this.

(1) FNN construction. For a selected (either Mamdani or T-S) FIS, e.g. FIS(TXMP,ATP;TGMP), construct its FNN, which is similar to that given in Figure 3. This is automatically done in FCPN tool by generating its Matlab code.

(2) Data acquisition. We then need to acquire the input and output data for training the FNN. For illustrative purposes, we obtain the data by running the above mentioned CPN/ODE model in FCPN tool and exporting the data.

(3) Data preprocessing. Take FIS(TXMP,ATP;TGMP) as an example. We need to compute the concentration change (increment or decrement) of an output, e.g. TGMP, resulting from the inputs, e.g. TXMP and ATP, at each selected time step.

(4) Training and testing. We then feed the preprocessed data into the FNN and train it in Matlab, which implicitly achieves learning the fuzzy rules of a FIS, e.g. FIS(TXMP,ATP;TGMP). After testing, we will obtain values of all fuzzy parameters of membership functions and fuzzy rules of the FIS. The training and testing results of FIS(TXMP,ATP;TGMP) together with their errors are given in Figure 10. The FNN codes of the example can be found in the FCPN tool website (see https://github.com/liufei2016/fcpn/blob/master/Examples/Example_FNN.zip), and the detailed procedure for learning fuzzy rules is given in our tool manual [38].

(5) Fuzzy parameter determination. When we obtain the fuzzy parameter values from learning the FNN, we need to add them to the FCPN model to obtain the final model with the FCPN tool.

Perform hybrid simulation

Hybrid simulation. After determining the structure and parameters of a model, we can now perform hybrid simulation to visually investigate its dynamic behaviour using the simulation

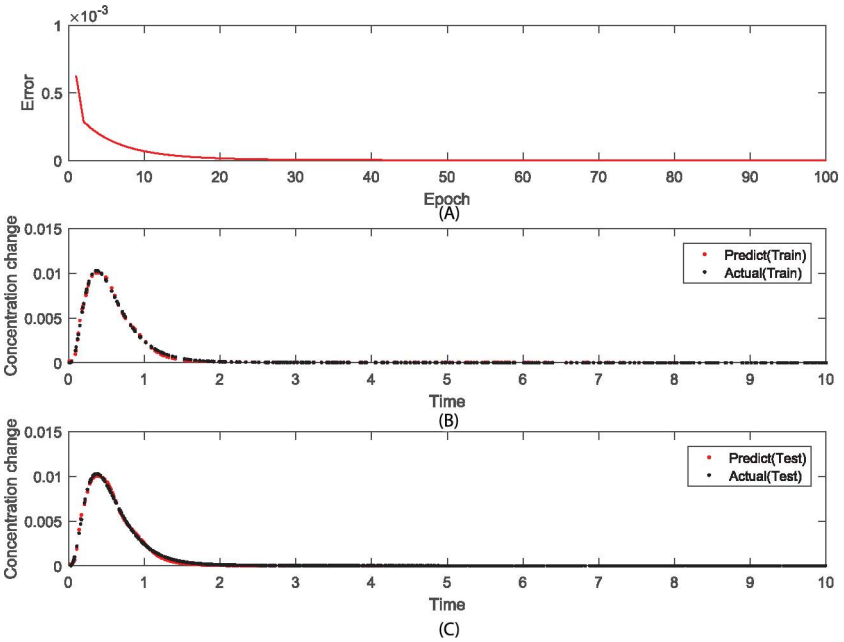

Figure 10. Training and testing result. (**A**) The variation of the errors calculated by Eq. 7 in 100 epoches. (**B**) Concentration changes of TGMP resulting from TXMP and ATP using the training dataset during the training. (**C**) Concentration changes of TGMP resulting from TXMP and ATP using the testing dataset during testing. The ratio of the training dataset to the testing dataset is 3:7.

algorithm given above and implemented in the FCPN tool. For hybrid models like FCPNs, due to their heterogeneous compositions with distinct modelling formalisms, we usually cannot directly perform mathematical or formal analysis of the models. In this case, simulation may be the best means to reveal the dynamic behaviour of a model.

In the FCPN tool, after setting the initial values for places, start and end of the simulation time and the simulation step, we can start simulation. After it finishes, the corresponding plot will be shown immediately.

Tuning fuzzy parameter with simulation. If an FCPN model contains membership functions directly specified by experts, we may have to run simulation and check simulation traces to tune each fuzzy parameter specified by hand. This step aims to refine the fuzzy parameters determined by experts in the previous stage. After multiple iterations between the parameter determining stage and the simulation stage, we can finally determine appropriate fuzzy parameters. Besides, in this step we may adopt all the analysis methods given in the next stage. To avoid repetition, we only describe them in the next section.

Result analysis and validation

Having finished the construction and simulation of a model, we have to analyse and validate it before utilizing it for prediction; otherwise, it may result in wrong predictions. Depending on whether reference data (i.e. real or measured data) is available or not, different methods can be adopted.

Objective validation. If reference data is available, we may adopt objective approaches to compare the simulation trace of a species with its corresponding reference trace in terms of such criteria as mean-square errors or absolute errors [39, 40].

In the following, we will take the model given in Figure 9 as an example to illustrate how to analyse an FCPN model in different fuzzy settings.

(1) Mamdani setting. First we only consider a Mamdani FIS in our model, i.e. FIS(TXMP,ATP;TGMP), by setting the other part of the model to the ODE semantics, and compare its behaviour with that of the corresponding pure CPN (ODE) model, illustrated in Figure 11A. From the plot, we can see there is a slight discrepancy between the hybrid (Mamdani) trace and the pure ODE trace (the mean squared error is 1.37661E-04), but both traces show a very similar trend of change for the species TIMP. Moreover, the hybrid trace gives more insight into the evolution of the species than the corresponding qualitative model, and therefore we should try to construct a hybrid model with FCPNs rather than a qualitative model, if there is not sufficient data available in order to better reveal the mechanism of the system to be studied. In fact, even if we do not resort to learning of fuzzy rules, and specify the rules and parameters only with experts' experience, we can obtain similar results as given in Figure 11A.

(2) T-S setting. We then only consider a T-S FIS in our model, i.e. FIS(TIMP,PP;TIMP), by setting the other part of the model to the ODE semantics, and compare its behaviour with that of the corresponding pure ODE model, illustrated in Figure 11B. From the plot, we can see that the two plots (fuzzy and ODE) do well match (with a mean squared error of 1.21706E-09). In fact, the T-S model usually gives a much more accurate result, if we learn the parameters with FNNs. However, it is not easy to specify the T-S fuzzy rules by hand.

(3) A mixture of Mamdani and T-S FISs. Finally, we combine these two kinds of fuzzy rules in one model to explore its dynamic behaviour; the comparison is given in Figure 11C. The fuzzy and ODE plots for each species are still very consistent, which shows that a mixed use of both kinds of fuzzy rules in one model is also applicable.

Subjective validation. However, for FCPN models, usually not all species can be measured to obtain their real traces and thus not all kinetic data are available; therefore, we may have to combine both objective and subjective approaches to evaluate the validity of models. In the subjective approach, experts usually are asked to check the simulation plots and examine whether the directions (i.e. the general tendency in terms of increasing or decreasing) of the output behaviours are correct or even whether

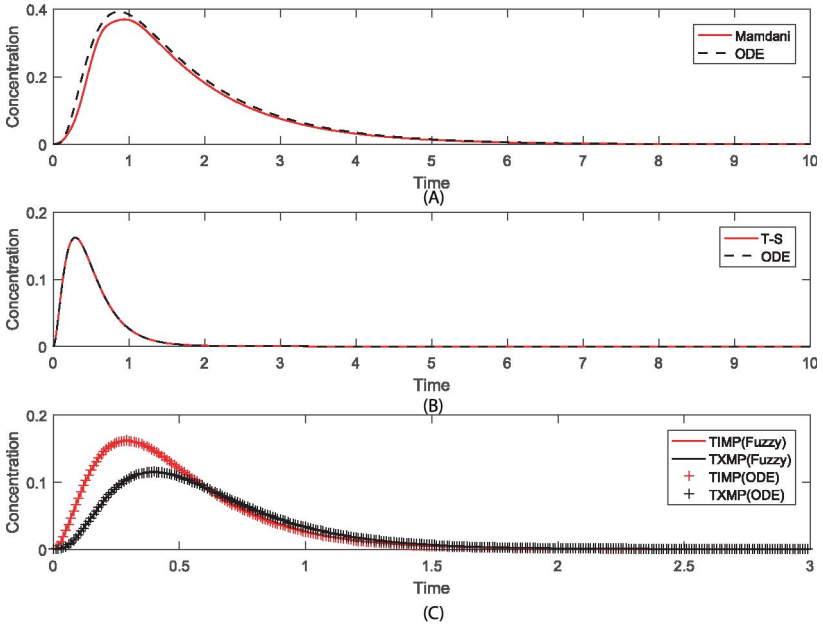

Figure 11. (A) ODE plot and fuzzy plot in the Mamadani setting for TGMP with a mean squared error of 1.37661E-04. (B) ODE plot and fuzzy plot in the T-S setting for TIMP with a mean squared error of 1.21706E-09. (C) ODE plot and fuzzy plot in both Mamadani and T-S setting for TXMP and TIMP with mean squared errors of 3.02501E-09 and 1.32112E-09, respectively.

the magnitudes are reasonable, as experts of a system usually know the directions and possibly the range of the magnitudes of the output behaviours [41].

Subjective validation aided by simulative model checking. Simulative model checking based on Probabilistic Linear-time Temporal Logic (PLTL), implemented in the MC2 tool [31], is used to determine if time series traces of a model fulfil given properties specified in temporal logic. With PLTL, we can check both qualitative properties like the general trend of the behaviour and quantitative properties over absolute concentration values. As described above, the power of FCPNs is to combine qualitative components with quantitative components in one model. Therefore, simulative model checking is an ideal tool to check both the qualitative and quantitative properties of an FCPN model.

For example, we can use the following formula to check if the concentration of a species x (e.g. TGMP) rises once and then strongly falls forever (the change trend of x):

$$P_{>=1}[F(d[\$x] > 0) \wedge (d[\$x] > 0 \ U \ (G(d[\$x] < 0)))]$$

where d[$x] returns the derivative of the concentration of a species x at each time point. Such a formula is very useful for checking an FCPN model with fuzzy rules directly specified by experts. Moreover with PLTL, we can at the same time check a number of traces for many species. This advantage of PLTL offers a good means for tuning fuzzy parameters when membership functions are manually specified by experts.

Discussion

FCPNs, by combining ODEs with FISs, offer a good means to model biological systems with uncertainties due to the lack of kinetic data or insufficient understanding of the mechanisms of biological systems. With FCPNs, some components are built as deterministic ODEs, if kinetic data are sufficient, and the others as uncertain FISs by resorting to the experience of biologists, if kinetic data are insufficient or unavailable. Therefore, FCPNs offer a tool for integrating experts' qualitative knowledge with measured quantitative data, and thus can partially overcome the challenges of integrated modelling of biological systems due to uncertainties.

Moreover, resorting to appropriate simulation algorithms, e.g. as given in this paper for the first time, and simulative model checking as given, e.g. in [42], we can observe and check the dynamic behaviour of the constructed qualitative/quantitative hybrid FCPN models. That is, not only the FCPN model can be graphically seen, but also the model's behaviour can be visually observed. Therefore, this permits biologists to check their qualitative knowledge through dynamic simulation runs.

This paper not only gives a formal definition and simulation algorithm of FCPNs, but also presents a detailed workflow comprising a set of methods and associated tools for how to utilize FCPNs to model and analyse biological systems. This workflow should clarify many modelling and analysis issues of FCPNs and offers an easy-to-follow guide for constructing hybrid qualitative and quantitative models. We have implemented the modelling and simulation of FCPNs in a tool, called FCPN tool, which can be freely accessed via the given website. We hope this will promote the wider application of FCPNs to address the integrated modelling challenges due to uncertainties, thus contributing to the systems biology area.

In a next step, we will continue to improve the FCPN tool according to the experience of users and to consider more functionalities by thoroughly analysing further integrated modelling challenges.

Key Points

- Integrated modelling of biological systems is challenged by composing components with sufficient kinetic data and components with insufficient kinetic data or components built only using experts' experience and knowledge.

- Fuzzy continuous Petri nets (FCPNs), by combining continuous Petri nets with fuzzy inference systems, offer a hybrid uncertain/certain approach to integrated modelling of such biological systems with uncertainties.
- In this paper, we give a formal definition of FCPNs and a corresponding simulation algorithm of FCPNs, and focus on presenting a methodology and workflow utilizing FCPNs to achieve hybrid (uncertain/certain) modelling of biological systems.
- We hope this research will promote the wide application of FCPNs and address the uncertain/certain integrated modelling challenge in the systems biology area.

Funding

National Key R&D Program of China (2018YFC0830900); National Natural Science Foundation of China (61873094); Science and Technology Program of Guangzhou, China (201804010246); Natural Science Foundation of Guangdong Province of China (2018A030313338).

References

1. Kitano H. Systems biology: a brief overview. *Science* 2002; **295**(5560):1662–4.
2. Aderem A. Systems biology: its practice and challenges. *Cell* 2005;**121**(4):511–3.
3. Covert MW, Xiao N, Chen TJ, *et al*. Integrating metabolic, transcriptional regulatory and signal transduction models in Escherichia coli. *Bioinformatics* 2008;**24**(18):2044–50.
4. Karr JR, Sanghvi JC, Macklin DN, *et al*. A whole-cell computational model predicts phenotype from genotype. *Cell*, 2012;**150**(2):389–401.
5. Carrera J, Covert MW. Why build whole-cell models? *Trends Cell Biol* 2015;**25**(12):719–22.
6. Machado D, Costa RS, Rocha M, *et al*. Modeling formalisms in systems biology. *AMB Express* 2011;**1**(45).
7. Bardini R, Politano G, Benso A, *et al*. Multi-level and hybrid modelling approaches for systems biology. *Comput Struct Biotechnol J* 2017;**15**:396–402.
8. Balaban M, Hester P, Diallo S. Towards a theory of multimethod M&S approach: part I. In: *Proceedings of 2014 Winter Simulation Conference (WSC)*, Dec 2014, 1652–63.
9. Liu F, Heiner M, Gilbert D. Fuzzy Petri nets for modelling of uncertain biological systems. *Brief Bioinform* 2018; 1–10. https://doi.org/10.1093/bib/bby118.
10. Gilbert D, Heiner M, Lehrack S. A unifying framework for modelling and analysing biochemical pathways using Petri nets. In: Calder M, Gilmore S (eds). *Proc. 6th International Conference on Computational Methods in Systems Biology, Vol. 4695 of LNCS, 200–16*. Berlin Heidelberg: Springer, 2007.
11. Heiner M, Gilbert D, Donaldson R. *Petri Nets for Systems and Synthetic Biology, Vol. 5016 of LNCS, 215–264*. Springer, 2008.
12. Gillespie DT. Exact stochastic simulation of coupled chemical reactions. *J Phys Chem* 1977;**81**(25):2340–61.
13. Herajy M, Heiner M. Hybrid representation and simulation of stiff biochemical networks. *Nonlinear Anal Hybri Syst* 2012;**6**(4):942–59.
14. Matsuno H, Tanaka Y, Aoshima H, *et al*. Biopathways representation and simulation on hybrid functional Petri net. *Silico Biol* 2003;**3**(3):389–404.
15. Windhager L. Modeling of dynamic systems with Petri nets and fuzzy logic. PhD thesis. Fakultät für Mathematik, Informatik und Statistik, LMU München, April 2013.
16. Bordon J, Moškon M, Zimic N, *et al*. Semi-quantitative modeling of gene regulatory processes with unknown parameter values using fuzzy logic and Petri nets. *Fundamenta Informaticae* 2018;**160**(1–2):81–100.
17. Zadeh LA. Fuzzy sets. *Inform Control* 1965;(8):338–53.
18. Wang PP, Ruan D, Kerre EE. *Fuzzy Logic*. Berlin Heidelberg: Springer-Verlag, 2007.
19. Mamdani EH. Application of fuzzy algorithms for control of simple dynamic plant. *Proc Inst Electr Eng* 1974;**121**(12): 1585–8.
20. Takagi T, Sugeno M. Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans Syst Man Cybern* 1985;**15**(1):116–32.
21. Shihabudheen KV, Pillai GN. Recent advances in neuro-fuzzy system: a survey. *Knowl Based Syst* 2018;**152**:136–62.
22. Jang J-SR. ANFIS: adaptive-network-based fuzzy inference system. *IEEE Trans Syst Man Cybern* May 1993;**23**(3): 665–85.
23. Horikawa S, Furuhashi T, Uchikawa Y. On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm. *IEEE Trans Neural Netw* 1992;**3**(5):801–6.
24. Petri CA. Kommunikation mit automaten. PhD thesis. Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962.
25. Murata T. Petri nets: properties, analysis and applications. *Proc IEEE* 1989;**77**(4):541–80.
26. Heiner M, Lehrack S, Gilbert D, *et al*. Extended stochastic Petri nets for model-based design of wetlab experiments. *Trans Comput Syst Biol XI* 2009;**LNBI 5750**:138–63.
27. Breitling R, Gilbert D, Heiner M, *et al*. A structured approach for the engineering of biochemical network models, illustrated for signalling pathways. *Brief Bioinform* 2008;**9**(5): 404–21. https://doi.org/10.1093/bib/bbn026.
28. Gilbert D, Heiner M, Jayaweera Y, *et al*. Towards dynamic genome scale models. *Brief Bioinform* 2017. online: October 13, 2017. https://doi.org/10.1093/bib/bbx096.
29. Mangan S, Alon U. Structure and function of the feed-forward loop network motif. *Proc Natl Acad Sci* 2003;**100**(21): 11980–5.
30. Heiner M, Schwarick M, Wegener J. Charlie-an extensible Petri net analysis tool. In: Devillers R, Valmari A (eds). *Proc. PETRI NETS 2015, Vol. 9115 of LNCS, 200–11*. Springer, 2015.
31. Donaldson R, Gilbert D. A model checking approach to the parameter estimation of biochemical pathways. In: Heiner M, Uhrmacher AM (eds). *Computational Methods in Systems Biology*. Berlin, Heidelberg: Springer, 2008, 269–87.
32. Lavrova AI, Postnikov EB, Zyubin AY, *et al*. *ODE and random Boolean networks in application to modelling of 6-mercaptopurine metabolism*. 2016. http://arxiv.org/abs/1611.00054.
33. Panetta JC, Evans WE, Cheok MH. Mechanistic mathematical modelling of mercaptopurine effects on cell cycle of human acute lymphoblastic leukaemia cells. *Br J Cancer* 2006;**94**(1):93–100.
34. Soliman S, Heiner M. A unique transformation from ordinary differential equations to reaction networks. *PLoS One* 2010;**5**(12):e14284.
35. Zhan C, Yeung LF. Parameter estimation in systems biology models using spline approximation. *BMC Syst Biol* Jan 2011;**5**(1):14.

36. Ashyraliyev M, Fomekong-Nanfack Y, Kaandorp JA, *et al*. Systems biology: parameter estimation for biochemical models. *FEBS J* 2009;**276**.
37. Moles CG, Mendes P, Banga JR. Parameter estimation in biochemical pathways: a comparison of global optimization methods. *Genome Res* 2003;**13**(11):2467–74.
38. Liu F and Sun W. *Manual for FCPN–a fuzzy continuous Petri net modeling and simulation tool*. https://github.com/liufei2016/fcpn/blob/master/Manual.pdf.
39. Anderson J, Papachristodoulou A. On validation and invalidation of biological models. *BMC Bioinform* 2009;**10**(1):132.
40. Hasdemir D, Hoefsloot HCJ, Smilde AK. Validation and selection of ode based systems biology models: how to arrive at more reliable decisions. *BMC Syst Biol* 2015;**9**(1):32.
41. Sargent RG. Verification and validation of simulation models. In: *Proceedings of the 2011 Winter Simulation Conference (WSC)*, 2011, 183–98.
42. Gao Q, Gilbert D, Heiner M, *et al*. Multiscale modelling and analysis of planar cell polarity in the drosophila wing. *IEEE/ACM Trans Comput Biol Bioinform* 2013;**10**(2):337–51.