FCPN – A Fuzzy Continuous Petri Net Modeling and Simulation Tool

Fei Liu, Wujie Sun
School of Software Engineering,
South China University of Technology

March 5, 2019

Contents

1 Introduction **4**
- 1.1 Petri nets . . . 4
- 1.2 Continuous Petri nets . . . 4
- 1.3 Fuzzy logic . . . 5
  - 1.3.1 Mamdani fuzzy inference . . . 5
  - 1.3.2 T-S Fuzzy Inference . . . 8
- 1.4 Fuzzy continuous Petri nets . . . 8
- 1.5 FCPN tool . . . 8
- 1.6 Features - overview . . . 9
  - 1.6.1 Features for modeling . . . 10
  - 1.6.2 Features for simulation . . . 10

2 Modeling **11**
- 2.1 An overview of the main interface . . . 11
- 2.2 Draw a net . . . 11
  - 2.2.1 Edit a place . . . 12
  - 2.2.2 Edit a transition . . . 12
  - 2.2.3 Edit an arc . . . 13
  - 2.2.4 Use FIS . . . 14

3 Simulation **24**
- 3.1 Run simulation . . . 24
- 3.2 Show simulation results . . . 24
- 3.3 Export simulation results . . . 26
- 3.4 Rich functions . . . 26

4 Examples **28**
- 4.1 1D Diffusion Reaction . . . 28
  - 4.1.1 Introduction . . . 28
  - 4.1.2 Model . . . 28
  - 4.1.3 Simulation result . . . 36
- 4.2 Enzymatic reaction . . . 39
  - 4.2.1 Introduction . . . 39
  - 4.2.2 Model . . . 39
  - 4.2.3 Simulation result . . . 41
- 4.3 RKIP pathway . . . 43
  - 4.3.1 Introduction . . . 43
  - 4.3.2 Model . . . 43
  - 4.3.3 Simulation result . . . 43
- 4.4 6-mercaptopurine metabolism . . . 47
  - 4.4.1 Introduction . . . 47
  - 4.4.2 Model . . . 47
  - 4.4.3 Simulation result . . . 47

5 Combine FCPN with neuro-fuzzy system **54**
5.1 Introduction 54
5.2 Instruction 54
5.2.1 Output the file 54
5.2.2 Train datas in MATLAB 54

1 Introduction

Systems biology [1] aims to study the interactions between the components of a biological system and how these interactions cause the behaviour of the system as a whole. Modelling and simulation plays an essential role in the study of systems biology by employing the data to build mathematical or computational models of a biological system, which help biologists to better understand and predict the system. However, the complexity of biological systems gives rise to a number of modelling challenges, one of which is uncertainty [2]. The uncertainty further constitutes the structural uncertainty and parametric uncertainty for a biological model.

In the deterministic quantitative modeling aspect, continuous Petri nets (CPNs) [3] offer a graphical way to represent systems of ordinary differential equations (ODEs). In a CPN model, places take real-valued tokens to represent the concentration of species, while transitions represent continuous changes of concentrations. However, neither ODEs or CPNs can address the uncertainty modeling issue.

By combining fuzzy inference systems [4, 5] with CPNs, we further obtain fuzzy continuous Petri nets (FCPNs), which achieve the uncertainty modeling capability from fuzzy logic. FCPNs lie in the second category of fuzzy Petri nets, given in [6]. Therefore an FCPN model can be divided into two parts: the certain ODE part and the uncertain fuzzy reasoning part, thus achieving the modeling of a biological system where we have sufficient data for some components but insufficient information for others.

1.1 Petri nets

Petri nets [7] provide a formal and graphical representation of systems based on their firm mathematical foundation for the analysis of system properties. Petri nets are one of several mathematical modeling languages for the description of distributed systems. A Petri net is a directed bipartite graph, in which the nodes represent transitions (i.e. events that may occur, represented by bars) and places (i.e. conditions, represented by circles). The directed arcs describe the flow of places to transitions or vice versa. No arc is allowed between two places or two transitions. Each arc has its own weight, and the default value is 1.

Let's take the well-known chemical reaction, $2H_2 + O_2 \rightarrow 2H_2O$, as an example [7] to illustrate Petri nets, which is shown in Figure 1. Circles named H_2, O_2, and H_2O are places, while the square named t is a transition. H_2 and O_2 are preconditions for t while H_2O is the postcondition for t. The tokens of H_2, O_2, and H_2O are set to 30, 20, and 0, respectively.

1.2 Continuous Petri nets

A continuous Petri net (CPN) [3] is a variant of Petri nets. In a CPN model, the values identified on places are non-negative real values (no longer required to be non-negative integer values), which usually model the concentrations of species. In fact, a CPN offer a graphical representation of a system of ODEs.

Let us continue to use the chemical reaction, $2H_2 + O_2 \rightarrow 2H_2O$ as an example to illustrate CPNs. The structure of the CPN model is exactly like

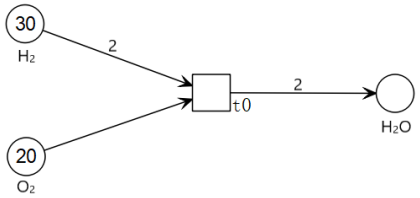


Figure 1: A Petri net example that models the chemical reaction, $2H_2 + O_2 \rightarrow 2H_2O$.

the one given in Figure 1. Besides, we associate a kinetic rate, e.g., 0.1, with the transition t by adopting a mass action law, represented as MassAction(0.1). Here MassAction() is a macro that generates the rate function of a transition using its preplaces and taking a parameter as its argument. For example, MassAction(0.1) for transition t generates a rate function, $0.1 * H_2^2 * O_2$. The set of ODEs underlying the CPN model given in Figure 1 is shown in Table 1.

Table 1: ODEs of the CPN model given in Figure 1.

Species	ODE
H_2	$dH_2/dt = -2 * (0.1 * H_2^2 * O_2)$
O_2	$dO_2/dt = -1 * (0.1 * H_2^2 * O_2)$
H_2O	$dH_2O/dt = 2 * (0.1 * H_2^2 * O_2)$

1.3 Fuzzy logic

Fuzzy logic [4, 5] is a form of many-valued logic in which the truth values of variables may be any real number between 0 and 1. It is employed to handle the concept of partial truth, where the truth value may range between completely true and completely false. By contrast, in Boolean logic, the truth values of variables may only be the integer values 0 or 1. Fuzzy logic is based on the observation that people make decisions based on imprecise and non-numerical information, and fuzzy sets are mathematical means of representing vagueness and imprecise information.

Two widely used fuzzy inference systems are Mamdani and T-S inference systems. In the following, we will briefly introduce these two systems.

1.3.1 Mamdani fuzzy inference

Mamdani fuzzy inference [8] is one of the most commonly used fuzzy inference systems and was among the first control systems built using fuzzy set theory. It expects the output membership functions to be fuzzy sets, which are then defuzzified to obtain crisp output values. The diagram of the Mamdani fuzzy inference system is shown in Figure 2, which involves the following four components: fuzzification, fuzzy rule base, fuzzy inference engine and defuzzification. An example is also given in Figure 3 to illustrate the reasoning process of the Mamdani fuzzy inference system.

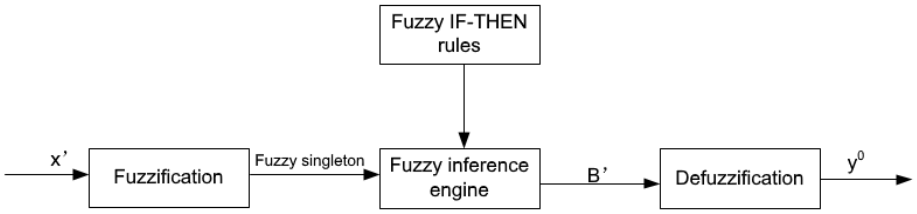


Figure 2: Diagram of the Mamdani fuzzy inference system.

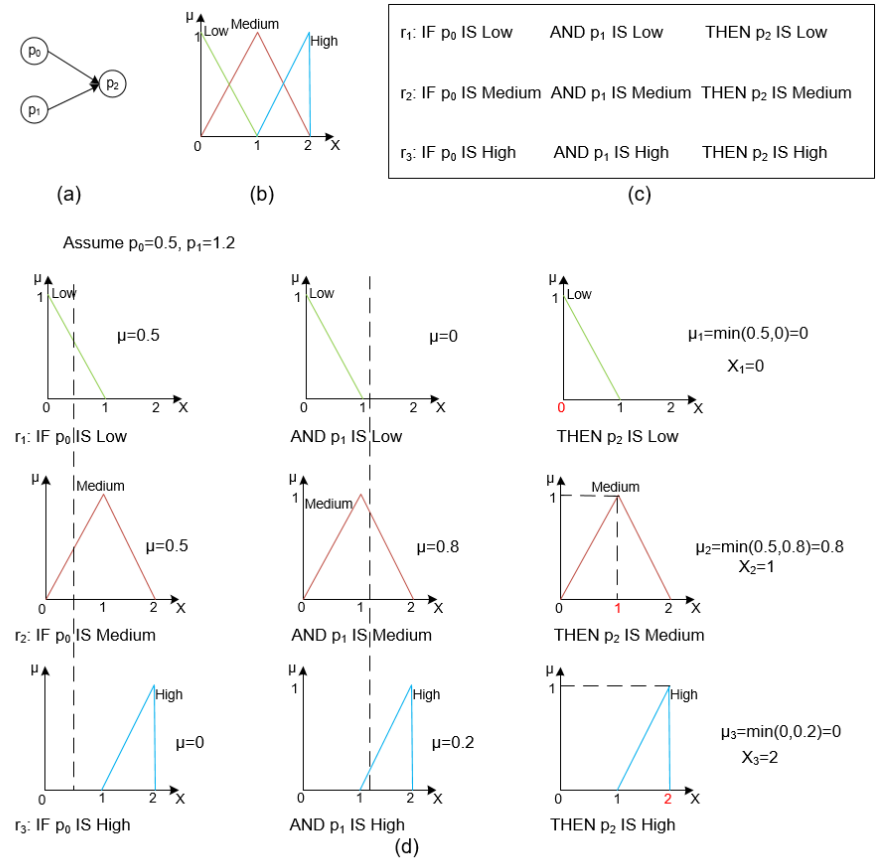


Figure 3: An example of the Mamdani fuzzy inference system.

(1)Fuzzification Fuzzification is used to fuzzify crisp input values into fuzzy values according to predefined membership functions for input variables. For

example, for the model given in Figure 3(a), p0 and p1 are inputs while p2 is the output. Three linguistic terms, Low, Medium and High, each taking a triangular fuzzy number, are defined for p0 to p2 (see Figure 3(b)). We assume that the minimum and maximum values of p0, p1 and p2 are 0 and 2, respectively. In the fuzzy membership functions shown in Figure 3(b), X indicates the value of each input and μ indicates the membership grade of the fuzzy set.

(2)Fuzzy rule base The rule base is a collection of fuzzy rules that are defined for a specific application. For example, for the model given in Figure 3(a), we can define the fuzzy rules given in Figure 3(c). A uniform format of the rules is:

$$r^r\text{: IF } x_1 \text{ IS } A_1^r \text{ AND ... AND IF } x_n \text{ IS } A_n^r \text{ THEN } y^r \text{ IS } B'^r.$$

(3)Fuzzy inference engine Fuzzy inference engine is used to execute all applicable rules in the rule base to compute the fuzzy output values. A detailed reasoning example is given in Figure 3(d).

(4)Defuzzification Defuzzification is used to defuzzify the fuzzy output values to get "crisp" output values. See an example in Figure 3(e).

1.3.2 T-S Fuzzy Inference

T-S fuzzy inference [9] is also called Sugeno or Takagi-Sugeno-Kang fuzzy inference. This method is similar to the Mamdani one except the difference that the T-S' outputs are either crisp linear expressions or constants rather than fuzzy values. The uniform format of the T-S rules is

$$r^r\text{: IF } x_1 \text{ IS } A_1^r \text{ AND ... AND IF } x_n \text{ IS } A_n^r \text{ THEN } y^r = a_1^r x_1 + ... + a_n^r x_n + c^r.$$

where a_i^r is ith coefficient in rth rule and c^r is a constant.

The process of the T-S fuzzy inference system is shown in Figure 4, which involves three components: fuzzification, fuzzy rule base, and fuzzy inference engine. An example is also given in Figure 5 to illustrate the reasoning process of the T-S fuzzy inference system.

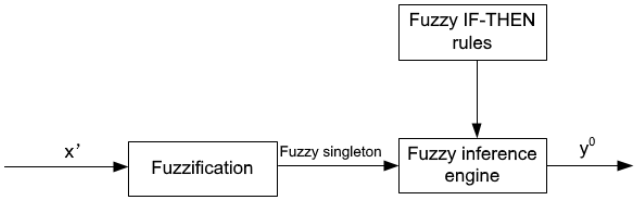


Figure 4: Diagram of the T-S fuzzy inference system.

1.4 Fuzzy continuous Petri nets

Adding fuzzy logic to continuous Petri nets gets fuzzy continuous Petri nets [6], which make the Petri nets more powerful. For example, if we want to get the value of H_2O using fuzzy logic, but ODE no longer works for H_2O, then the value of H_2O can be obtained by fuzzy logic.

Please note that when using fuzzy logic in fuzzy continuous Petri nets, the value we obtained is not the value of the variable, but the amount of change within the step. The change value of H_2O of current time step is obtained by the values of H_2 and O_2 of previous time step.

The result might be a little different from ODE. If the values of H_2 and O_2 are down to 18 and 14, the ODE result of H_2O is 12, while in fuzzy logic, the result of H_2O might be 11.8.

1.5 FCPN tool

The tool provides modeling and simulation functions of fuzzy continuous Petri nets for researchers in the field of systems biology. It includes three main functions: continuous Petri net modeling, fuzzy (Mamdani & T-S) modeling, and hybrid simulation. The aim of this software is to give an easy-to-use graphical tool for the construction and simulation of FCPN models.

We offer the Windows, Linux, and macOS (beta) versions of the software. To use the software, please follow the instruction in README.md and download the corresponding version of the tool.

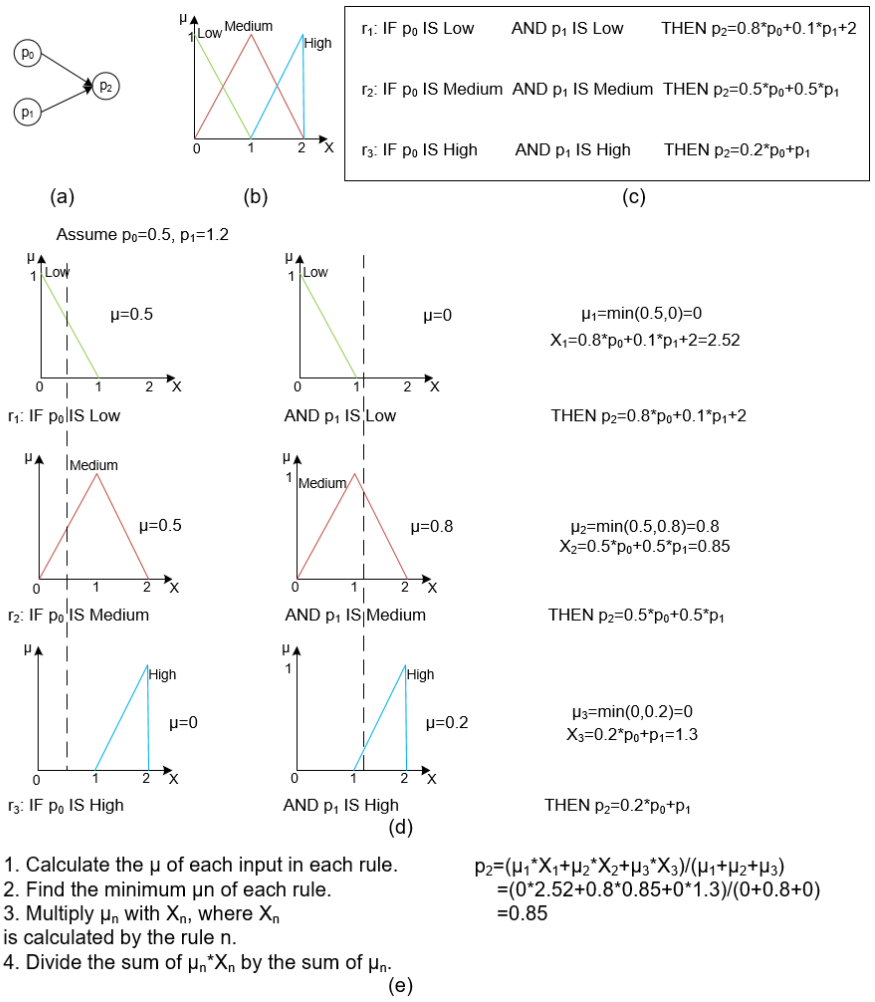


Figure 5: An example of the T-S fuzzy inference system.

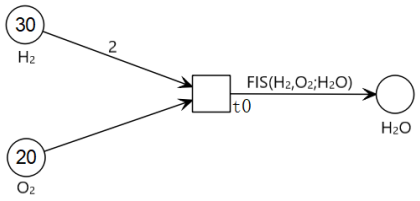


Figure 6: Example of fuzzy continuous Petri nets.

1.6 Features - overview

Before exploring all features in detail in the following sections, we will give a brief overview for the expected features here.

1.6.1 Features for modeling

- Concise and efficient interface design.
- Drawing of the Petri net graph as usual.
- Flexible user-defined functions.
- Multiple fuzzy logic choices.
- Simple and fast fuzzy logic settings.
- Save as Petri Net Markup Language (PNML) format
- Rich shortcut settings, such as undo, redo, save, print, etc.

1.6.2 Features for simulation

- Highly automated simulation process.
- Diversified export of simulation results.
- Custom simulation result drawing.

2 Modeling

This section will present a general step-by-step procedure for how to construct an FCPN model. We will take the chemical reaction $2H_2 + O_2 \rightarrow 2H_2O$ as an example [7] for the illustration of the procedure.

2.1 An overview of the main interface

Figure 7 shows the main interface of the software. The menu bar is on the top and the tool bar is on the left, which facilitates us to create the net conveniently. The tools from top to bottom include "new", "open", "save", "save as", "print", "undo", "redo", "delete a node", "normal cursor", "create a new place", "create a new transition", "create a new arc", and "start simulation".

The top window is used to draw the net and the bottom window to output a log. The bottom right slider can be used to zoom in or zoom out a net.

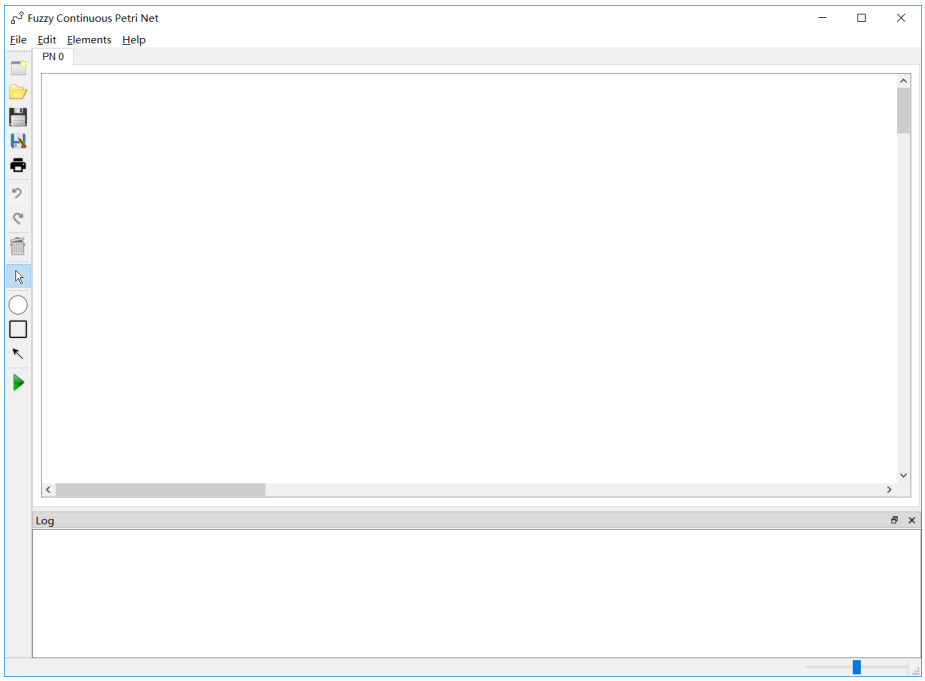


Figure 7: Main interface of the software.

2.2 Draw a net

We can draw a net using the following steps taking Figure 8 as an example:

- Left click the place button (the circle on the tool bar).
- Put three places on the palette where you want by left clicking the mouse.
- Left click the transition button (the square on the tool bar).

- Put one transition on the palette where you want by left clicking the mouse.
- Left click the arc button (the arrow on the tool bar).
- Left click a place, press the mouse and hold, move the mouse to a transition, and then release the mouse. In this way you can draw an arrow from a place to a transition or vice versa.
- Repeating these steps, an FCPN can be constructed; see Figure 8.

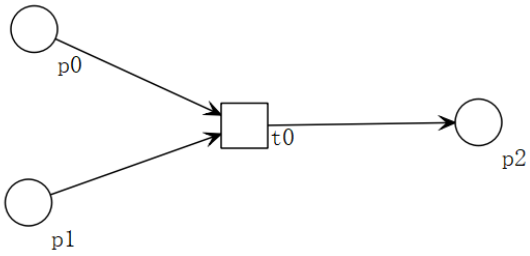


Figure 8: Draw the net.

2.2.1 Edit a place

We can edit the properties of a place in the following way:

- Double click the place on the palette.
- A window named Place Attributes (Figure 9) will show.
- Change the Name and Marking.
- After editing, left click the OK button.

For example, in Figure 8, we change the names of p0, p1, and p2 to H2, O2, and H2O, respectively, and change the markings of p0, p1, and p2 to 30, 20, and 0, respectively. Please note that the name of each place should be unique and a marking should be a non-negative real number.

2.2.2 Edit a transition

We can edit the properties of a transition in the following way:

- Double click the transition on the palette.
- A window named Transition Attributes (Figure 10) will show.
- Change the Name and Function.
- After editing, left click the OK button.

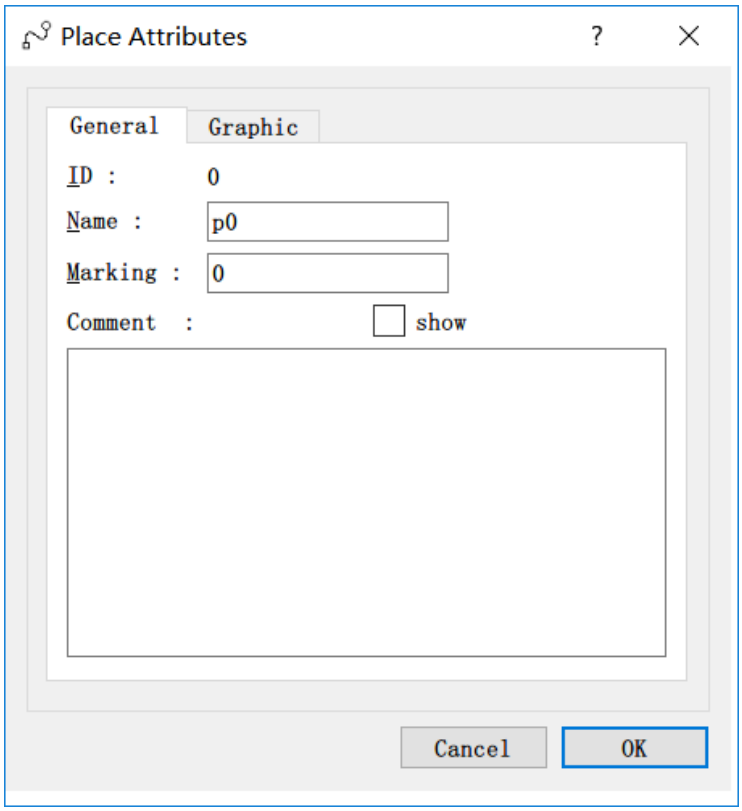


Figure 9: Edit a place.

For example, in Figure 8, we change the function of t0 from the default MassAction(1) to MassAction(0.1). Please note that the name of each transition should be unique and the function should be an expression like a constant "1", an expression containing its preplaces"0.1*H2", or a mass action "MassAction(0)". Remember not to change the "MassAction(1)" to "massaction(1)" or something like that as we distinguish the upper and lower case letters. And do not leave the function empty; otherwise a compile error will occur.

2.2.3 Edit an arc

We can edit th properties of an arc in the following way:

- Double click the arc on the palette.
- A window named Arc Attributes (Figure 11) will show.
- Change the Expression.
- After editing, left click the OK button.

For example, in Figure 8, we change the expression of arc from p0 (H2) to t0 to 2. Please note that the expression should be like "1", "H2", if you don't want

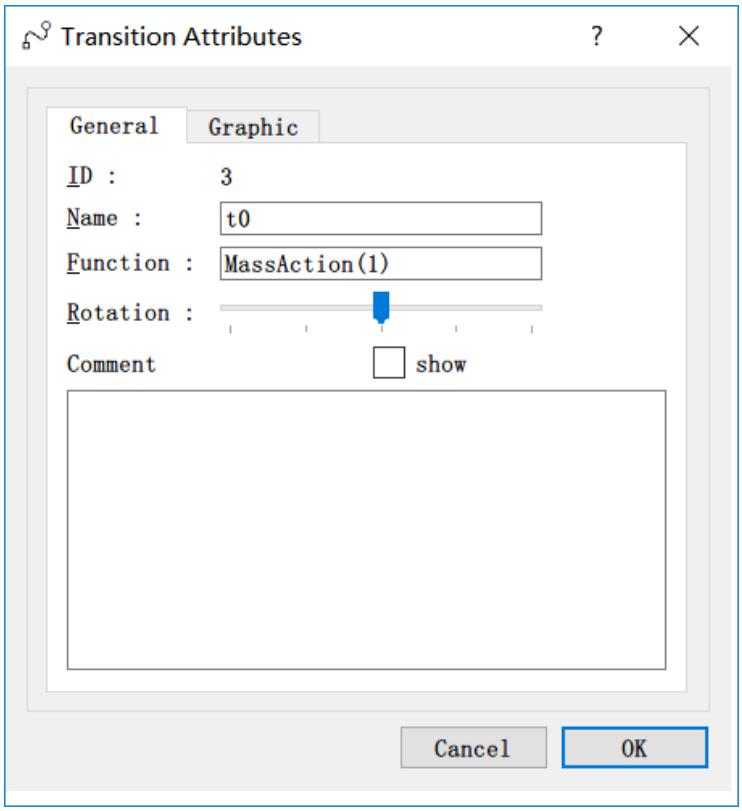


Figure 10: Edit a ransition.

use fuzzy logic, and do not leave the expression empty. Otherwise a compile error might occur.

2.2.4 Use FIS

FIS is the abbreviation of the fuzzy inference system. We provide two kinds of FISs: Mamdani and T-S. The steps are slightly different between Mamdani and T-S settings. Here are the steps for using Mamdani inference system:

- Double click the arc from t0 to p2 (H2O).
- Click the FIS editor button.
- Choose Mamdani FIS Type.
- Enter the time step and select the number of input variables.
- Click the Apply button.
- Choose each variable and its membership function type (either triangular or Gauss fuzzy numbers) in the table.
- Specify the minimum and maximum values for input variables.

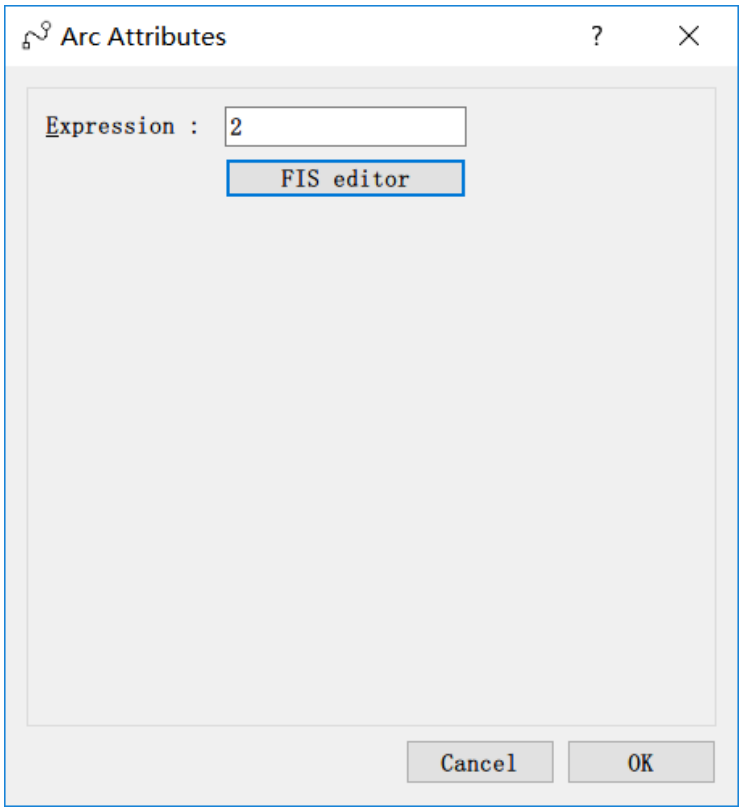


Figure 11: Edit an arc.

- Specify the minimum and maximum increment/decrement values at a time step for the output.

Make sure we only allow to specify one output variable whose type is PN_OUTPUT, and it should be the variable connected by the arc you chose. In this example, that's p2 (H2O). The values are shown in Figure 13.

Now we need to edit each variable's membership functions. Here are the steps:

- Click the Edit button on the first column.
- Choose the number of linguistic variables.
- Click the Apply button and the according label will show.
- For each function, enter the values of left/average, center/variance and right/-, which specify the parameters that are needed by triangular/Gauss fuzzy numbers.
- Click the Show Plot button to see whether it is the graph you want.
- Click the OK button to save the change and close the window.

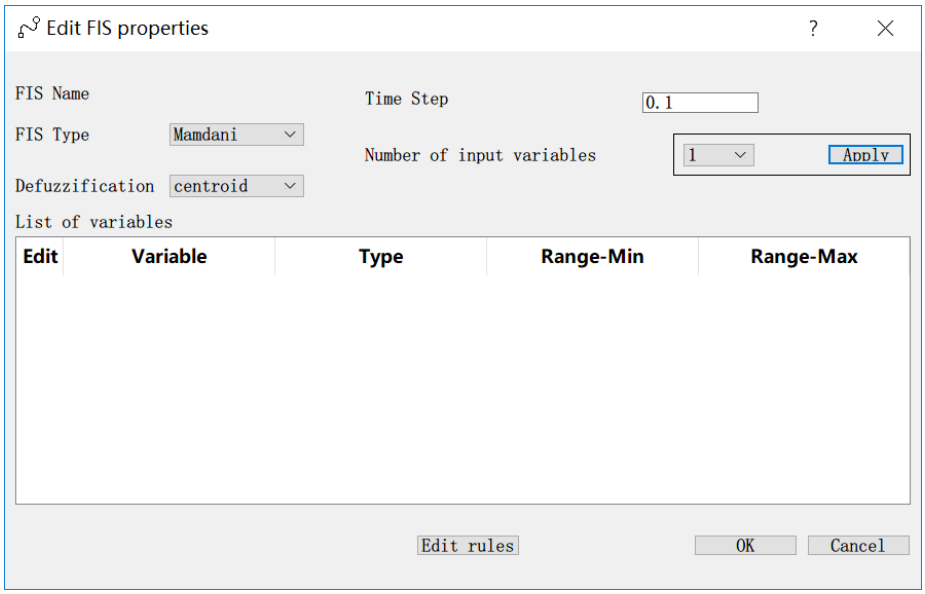


Figure 12: FIS editor.

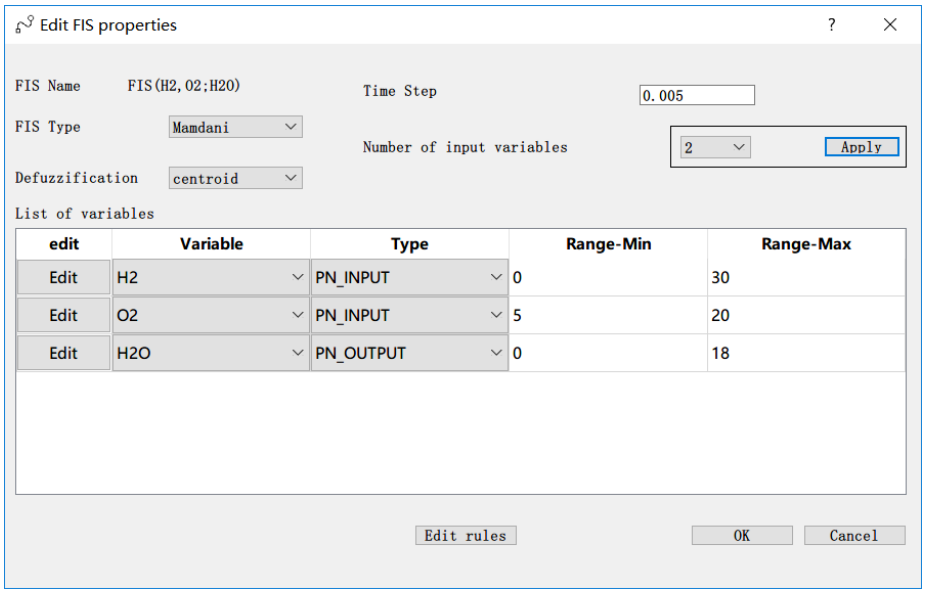


Figure 13: Edit variables.

- Repeat these steps to edit each variable.

We provide the user to choose the number (2, 3, 5, 7) of linguistic terms. The membership functions are shown in Figure 14, 15, and 16.

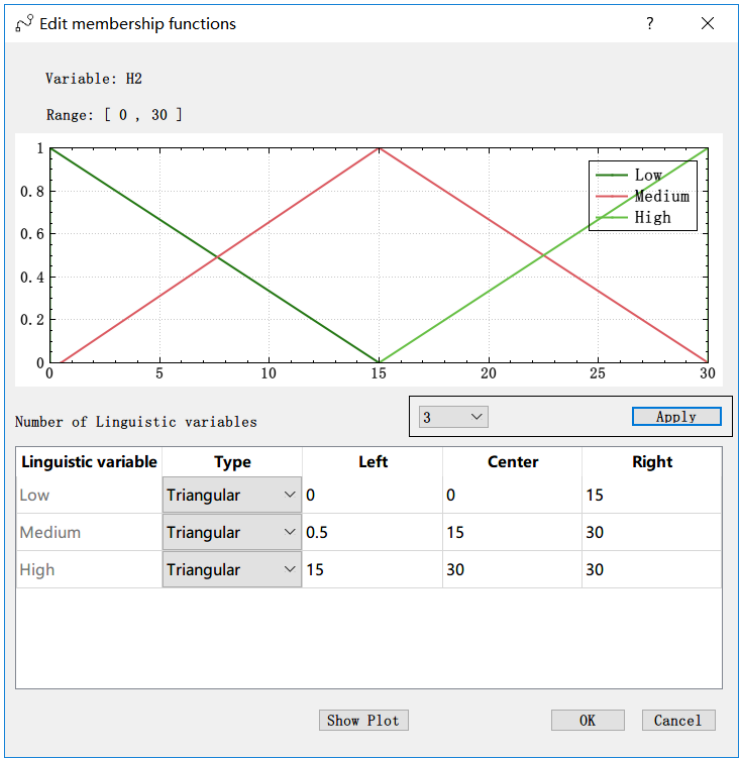


Figure 14: Membership functions of H2.

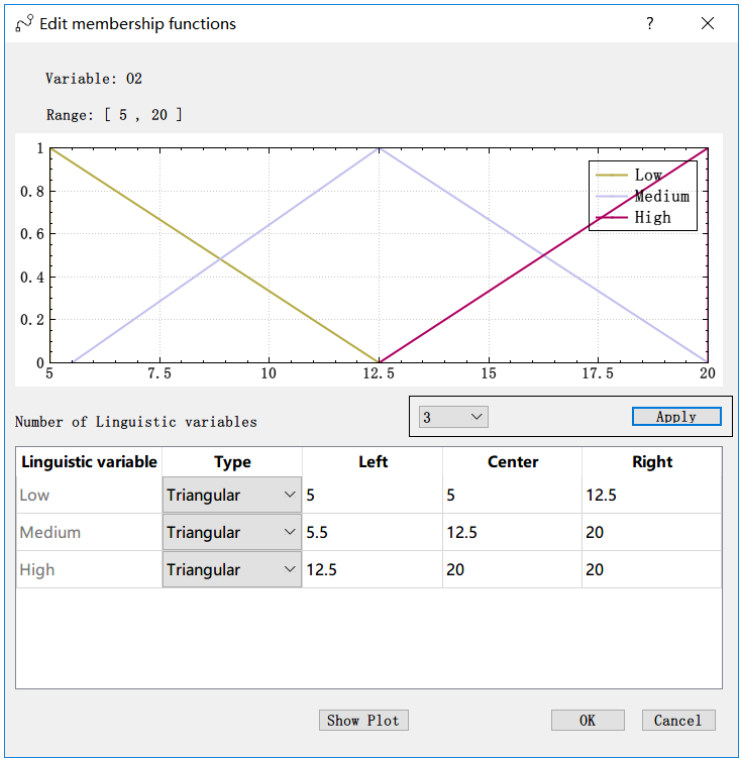


Figure 15: Membership functions of O2.

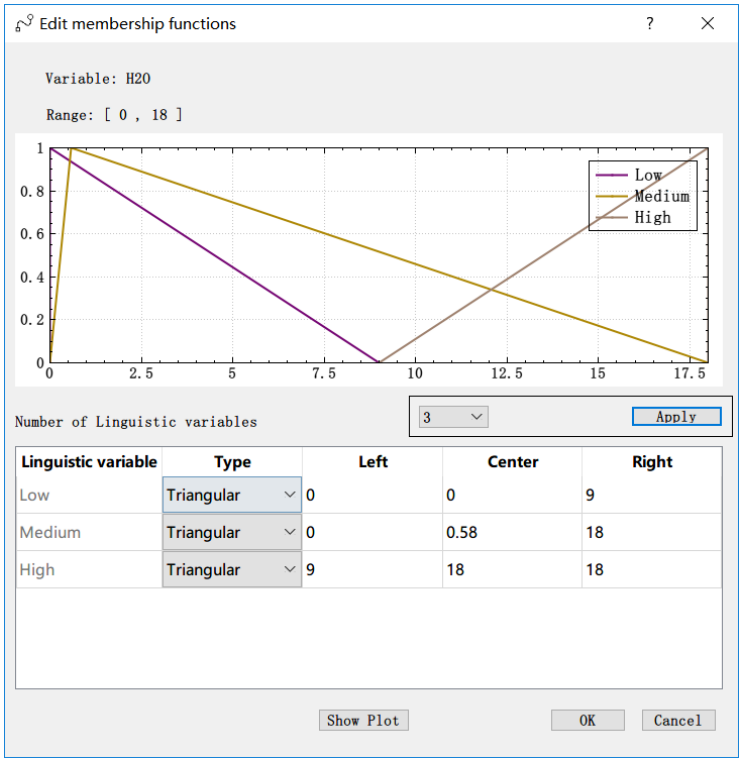


Figure 16: Membership functions of H2O.

Next we should edit rules by following these steps:

- Click the Edit rules button on the bottom.
- Select the items of a rule, and left click Add rule button to add it.
- Repeat the previous step to add rules.
- Click the OK button to save the changes and close the window.

We can select a row and left click Change rule button to change the rule you selected. Choosing a row and left clicking the Delete rule button can be used to delete the rule.

Make sure that the meanings of inputs' linguistic variables and output's linguistic variables are different. For example, as is shown in Figure 17, one of the rules is - IF H2 IS Low AND O2 IS Low Then H20 IS Low. This means that if the value of H2 is Low and the value of O2 is Low, then the increment of H2O is Low, not meaning that the value of H2O will be low.

The rules are shown in Figure 17.

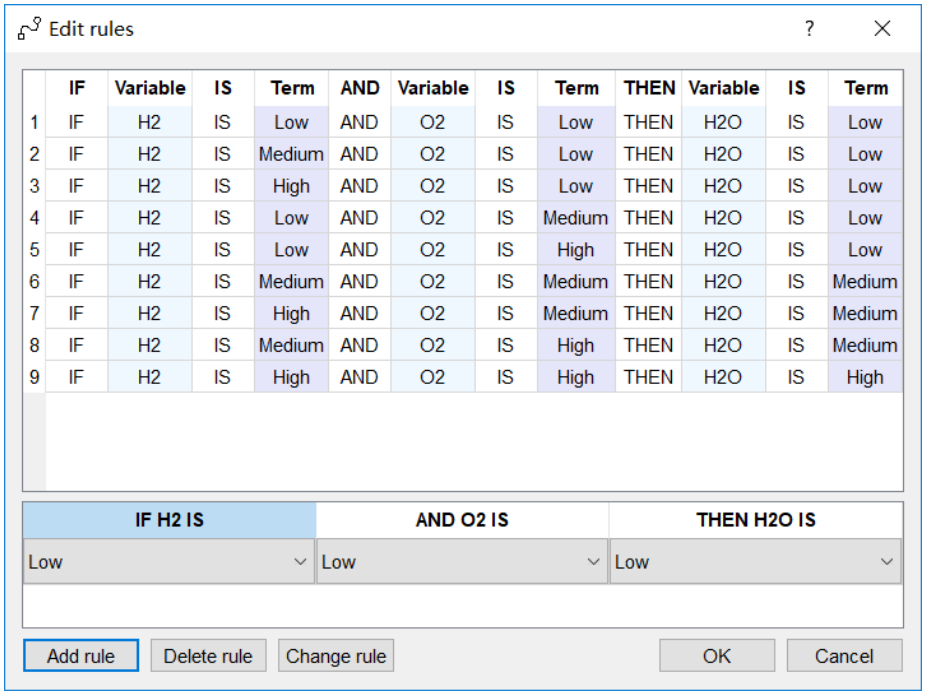


Figure 17: Edit rules.

Now left click the OK button of the FIS editor and left click the OK button of the arc to return to the main interface. The net should be like Figure 18.

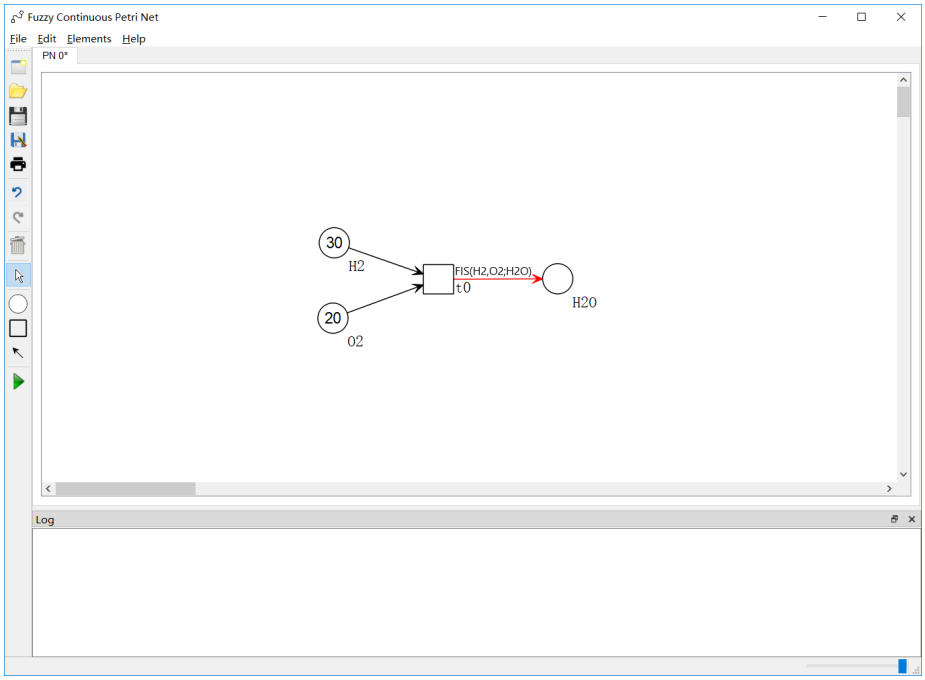


Figure 18: The constructed FCPN model with an FIS.

There are some differences between Mamdani and T-S settings.

First, in the T-S setting, when left clicking the Edit button of variable whose type is PN_OUTPUT, there is no need to edit its membership functions.

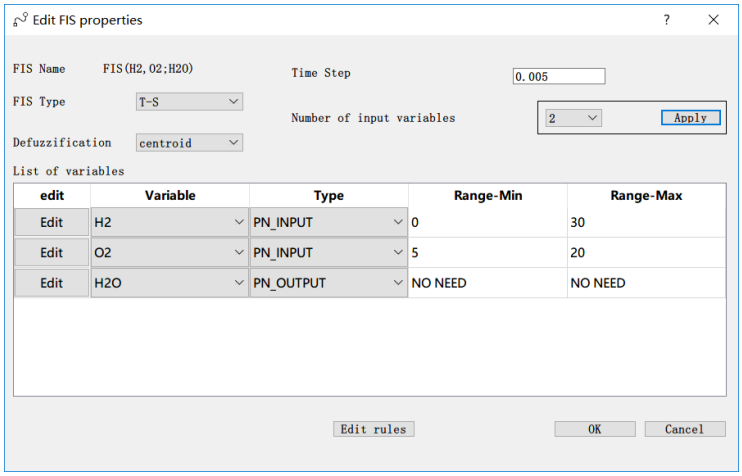


Figure 19: T-S FIS setting.

Second, when editing rules, the last column should be edited by the user as is shown in Figure 20. Add rules first and then change the content of the last column by double clicking the space of the last column of each row and entering the formulas. Then left click the OK button to save changes. Other steps are similar to the Mamdani setting.

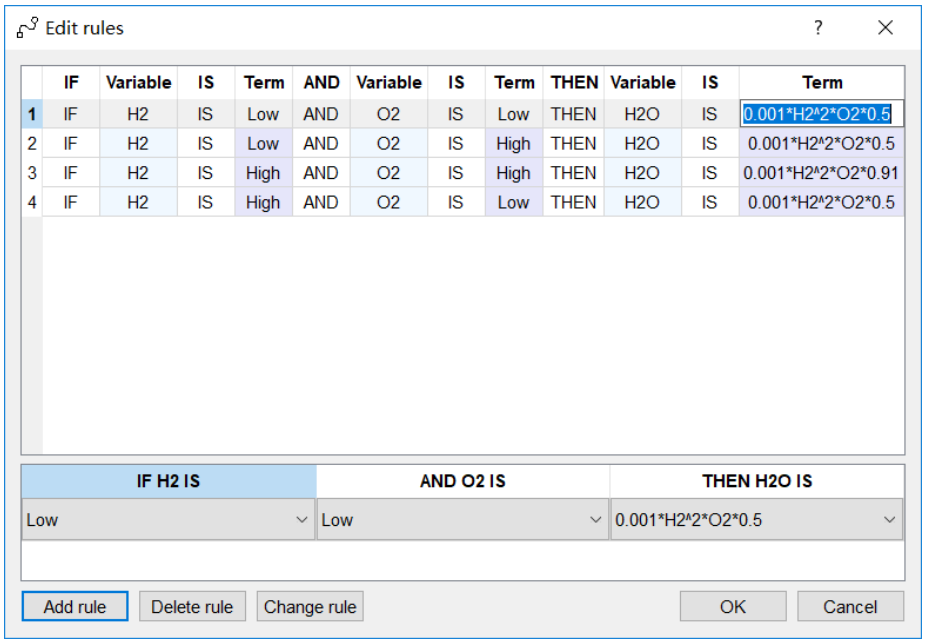


Figure 20: Enter rules when using T-S.

3 Simulation

After an FCPN is finished, left click the “start simulation” button on the tool bar to start the simulation function.

3.1 Run simulation

In the simulation dialogue shown in Figure 21, the user can first set simulation parameters, then left click the Start Simulation button to start a simulation. The settings include:

- Setting an interval start.
- Setting an interval start.
- Choosing an ODE solver (only one solver supported at present).
- Setting a time step.
- Choosing to check negative value or not.

Please make sure the time step is smaller than the step size given in the FIS setting. Otherwise the result might be incorrect.

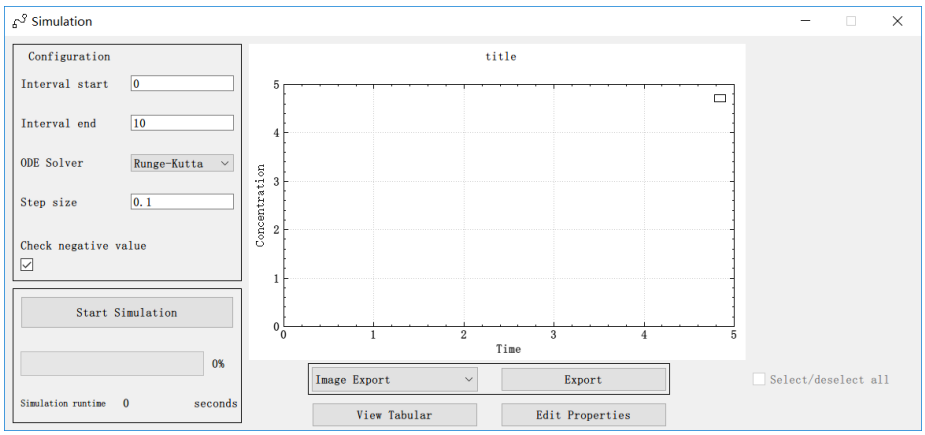


Figure 21: Simulation.

3.2 Show simulation results

During simulation, the progress bar shows the progress of a simulation. After simulation, the plot will show. The user can zoom in or zoom out to see the details and check or uncheck the boxes on the right to show or hide the according variables.

To avoid mis-operation, after entering the simulation, most of the functions in the main interface such as adding items, can not be used. So if you want to get back to edit in the main interface, remember to left click “normal cursor”

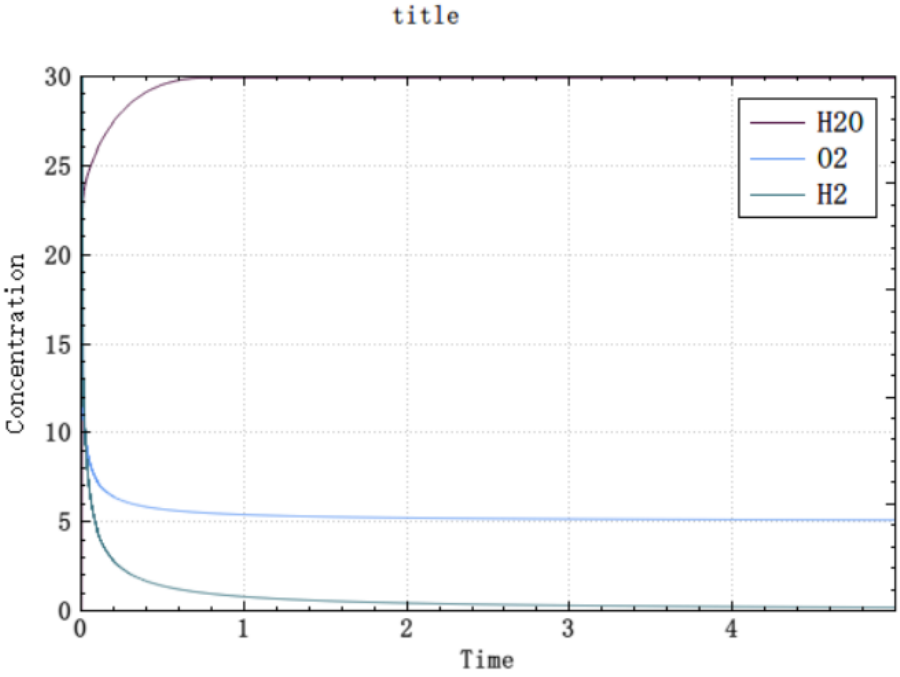


Figure 22: Simulation result (Mamdani).

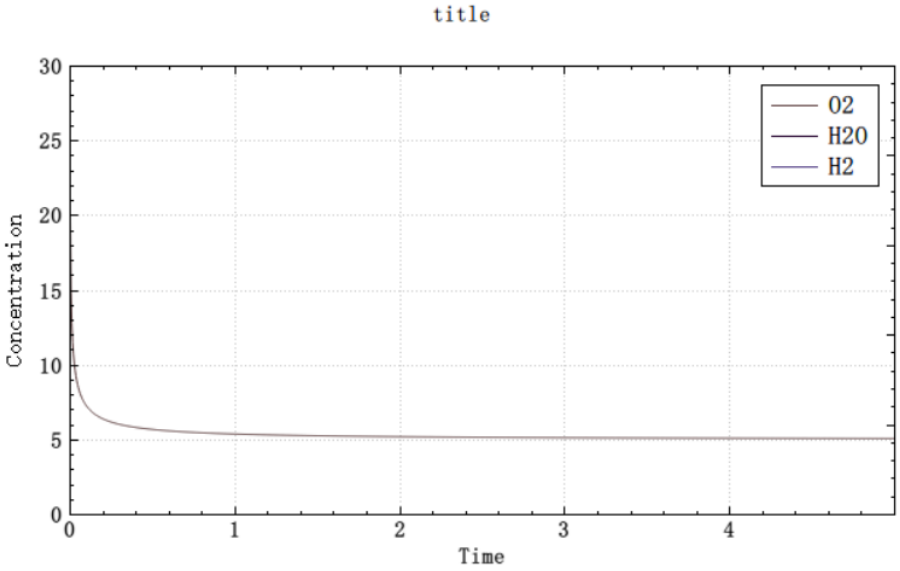


Figure 23: Only show the result of O2.

on the left tool bar. Otherwise you will find that most of the functions can't use.

Actually, the user can choose four modes: no FIS, Mamdani, T-S, and hybrid FIS, to run the simulation. Therefore, let us use the no FIS mode to show how to run the simulation.

- Close the simulation window.
- Left click "normal cursor".
- Delete the FIS arc and replace it with a new arc whose expression is 2.
- Start simulation again to see the result.

The result can be a little bit different, but still shows the general trend.

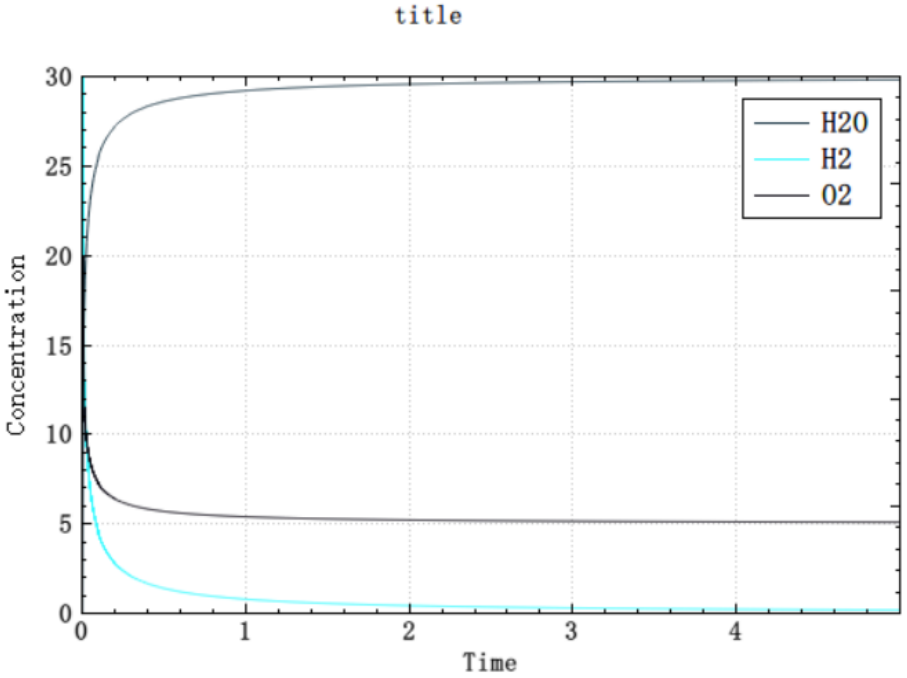


Figure 24: Simulation result (No FIS).

3.3 Export simulation results

The user can choose the export way and left click the Export button to export.

3.4 Rich functions

Different tools are provided which are located at the center of the bottom. The user can view the Tabular to see the values of variables by left clicking the View Tabular button or edit properties by left clicking the Edit Properties.

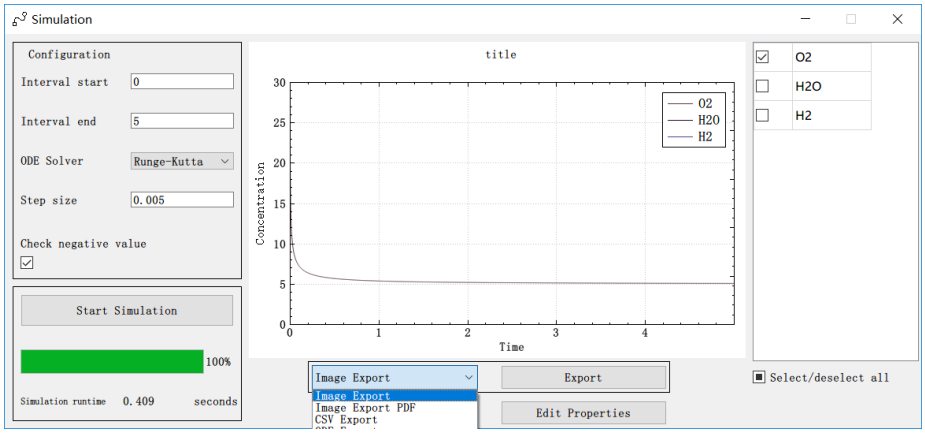


Figure 25: Export result.

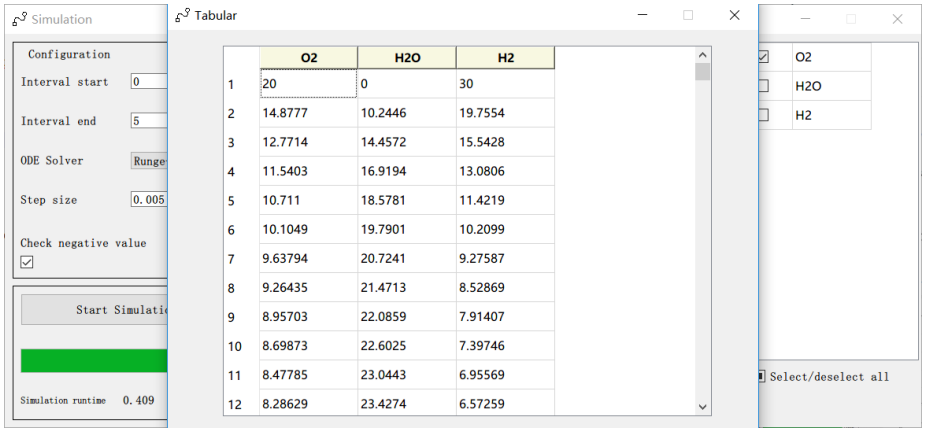


Figure 26: View Tabular.

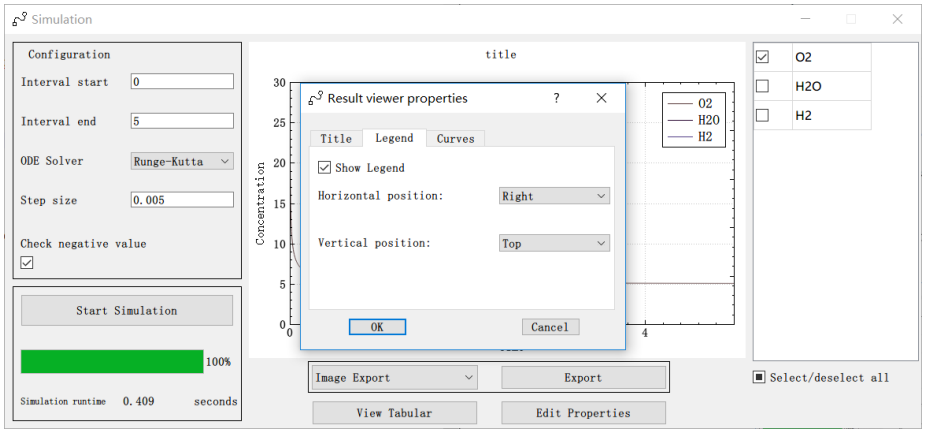


Figure 27: Edit properties.

4 Examples

In this section, we will show four examples to illustrate our tool. And we name the file of an example with A_B_C_D, where A is the name of the example, B is the FIS type we used, C is the interval end of the simulation, and D is the step size. The default interval start is 0 for a simulation.

The models of all the examples can be downloaded from Examples.

4.1 1D Diffusion Reaction

4.1.1 Introduction

The first example is one-dimensional (1D) diffusion reaction [10], which describes a species diffusing into one of its neighbour cells in a 1D grid.

4.1.2 Model

The model is shown in Figure 28.

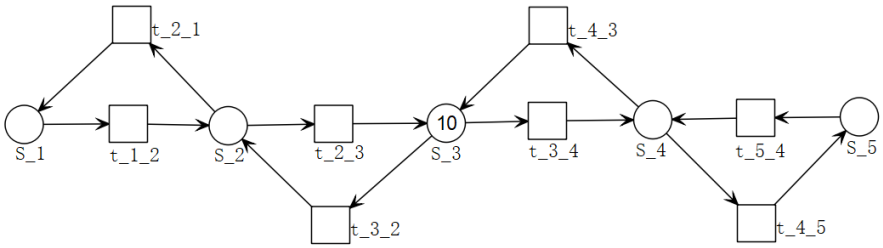


Figure 28: The model of 1D diffusion reaction.

Table 2: Transition functions of the model of 1D Diffusion Reaction.

Transition	Function
t_1_2	MassAction(1)
t_2_1	MassAction(1)
t_2_3	MassAction(1)
t_3_2	MassAction(1)
t_3_4	MassAction(1)
t_4_3	MassAction(1)
t_4_5	MassAction(1)
t_5_4	MassAction(1)

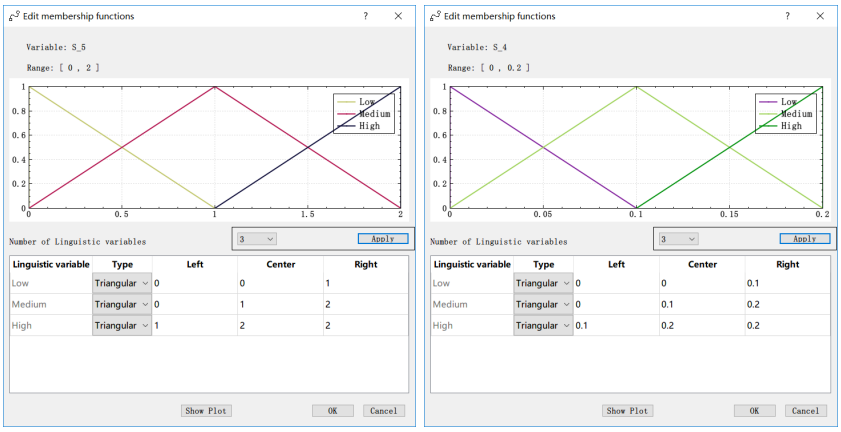


Figure 29: Membership functions of the 1D Diffusion Reaction model using the Mamdani setting.

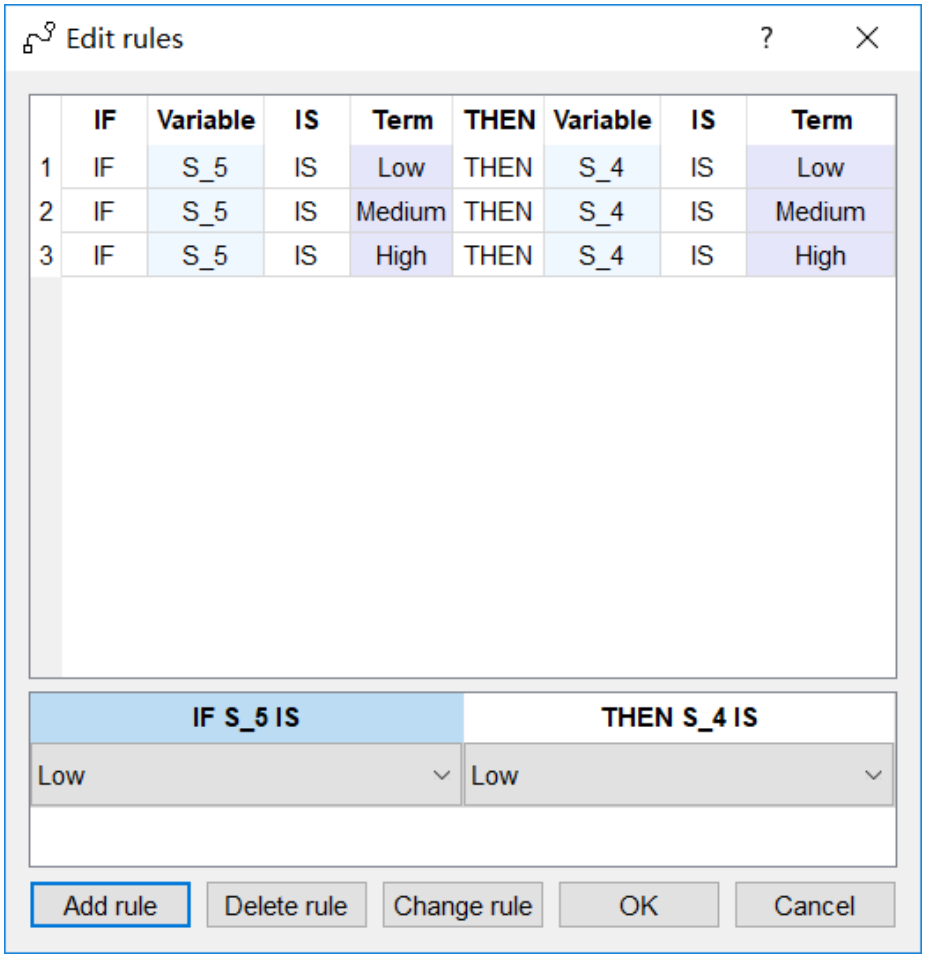


Figure 30: Rules of the 1D Diffusion Reaction model using the Mamdani setting.

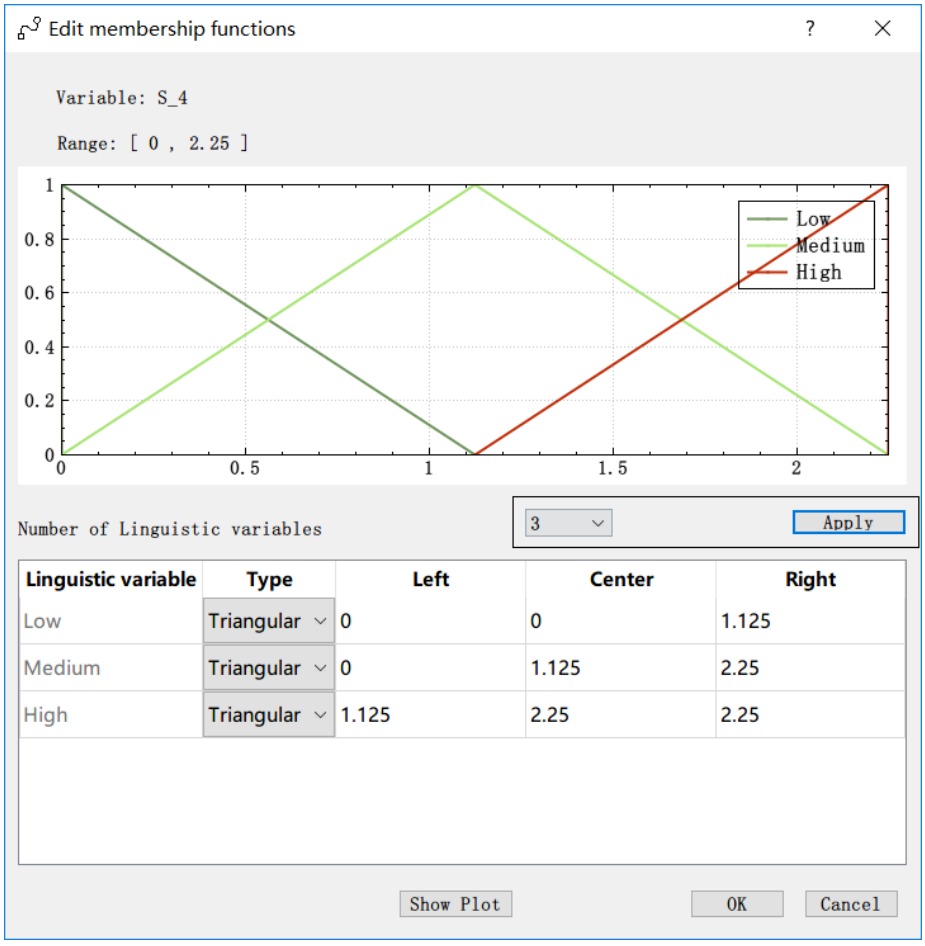


Figure 31: Membership functions of the 1D Diffusion Reaction model using the T-S setting.

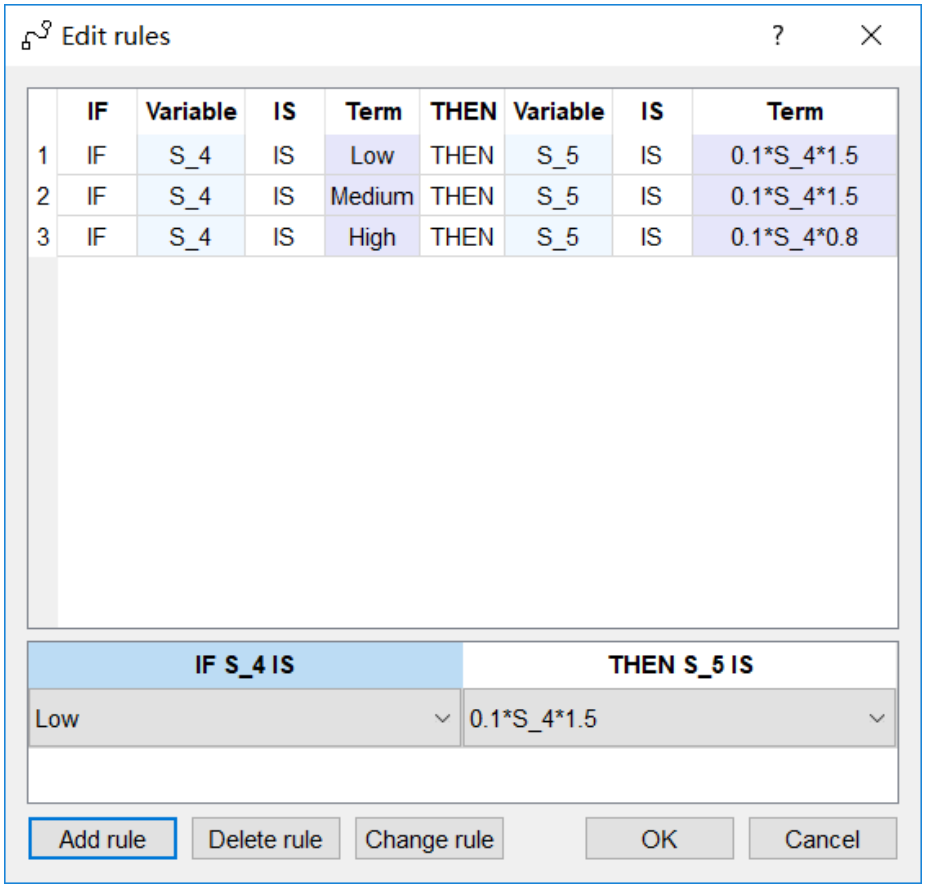


Figure 32: Rules of the 1D Diffusion Reaction model using the T-S setting.

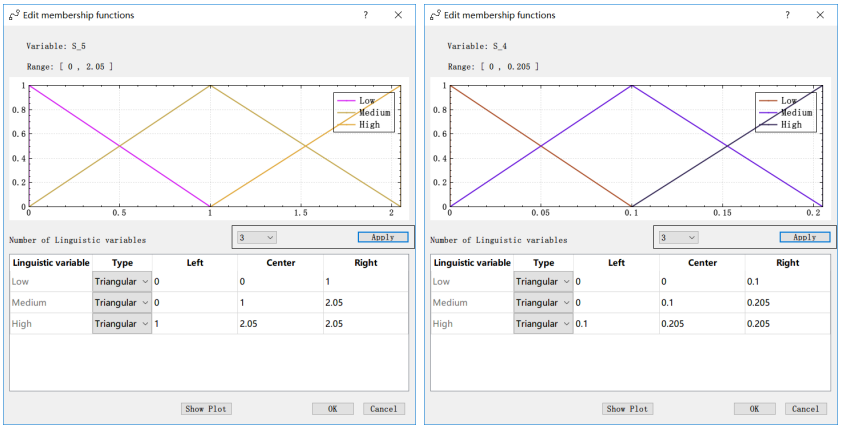


Figure 33: Membership functions of the 1D Diffusion Reaction model using the hybrid FIS setting (Mamdani).

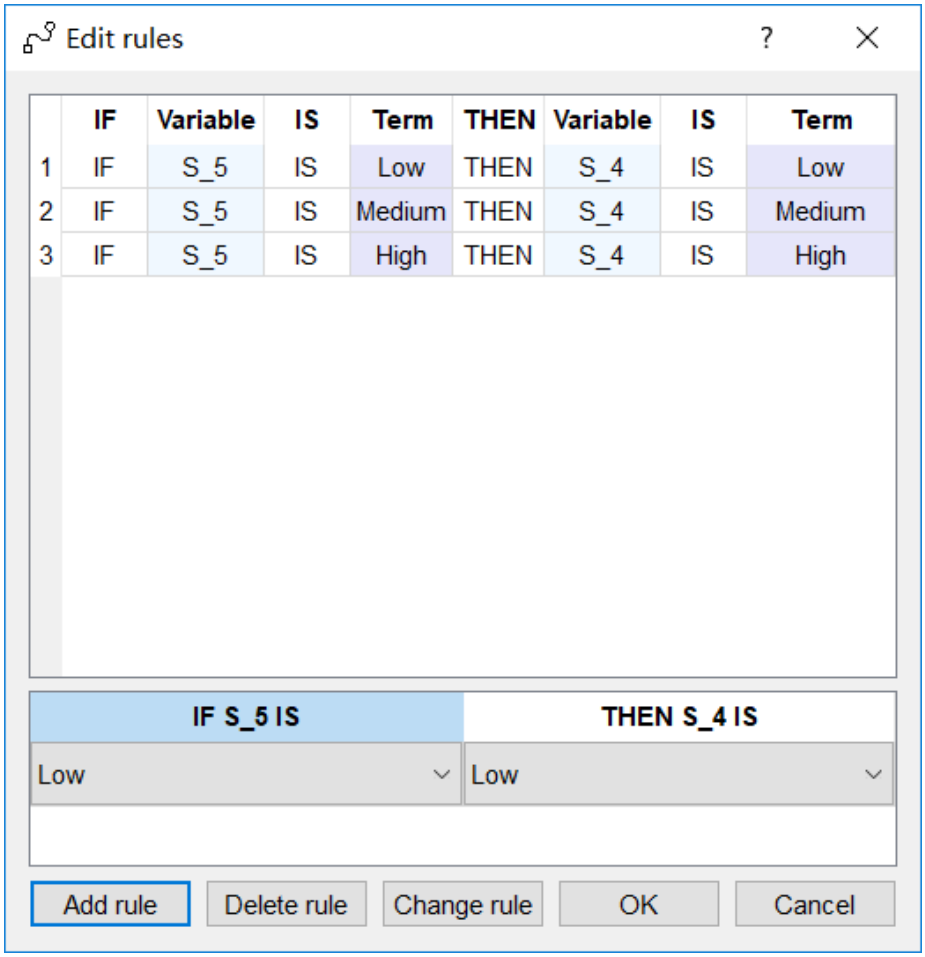


Figure 34: Rules of the 1D Diffusion Reaction model using the hybrid FIS setting (Mamdani).

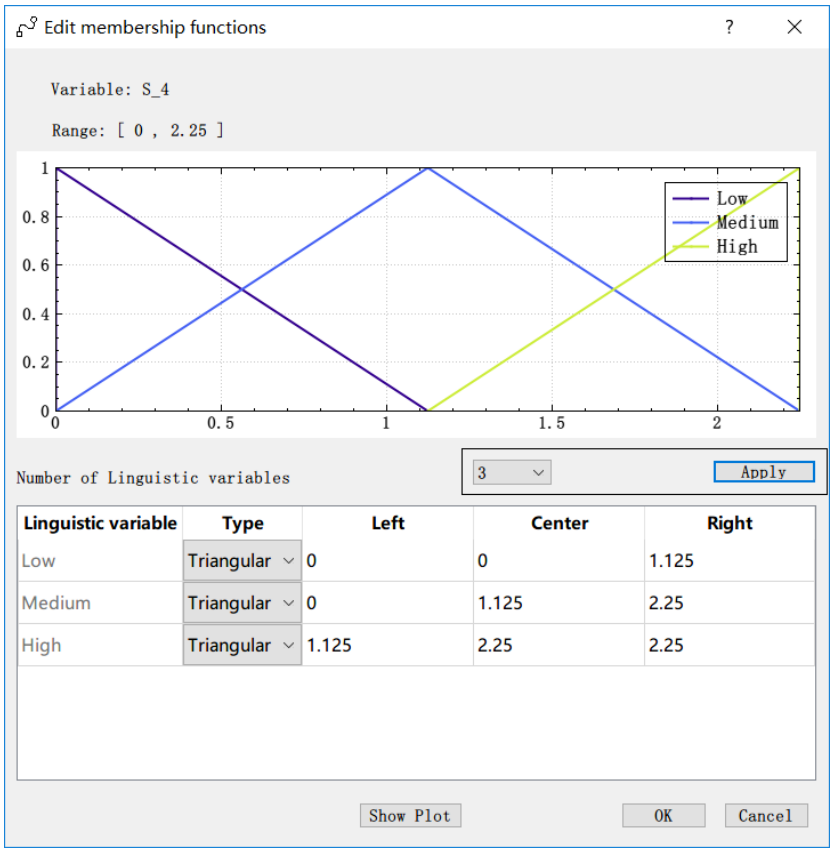


Figure 35: Membership functions of the 1D Diffusion Reaction model using the hybrid FIS setting (T-S).

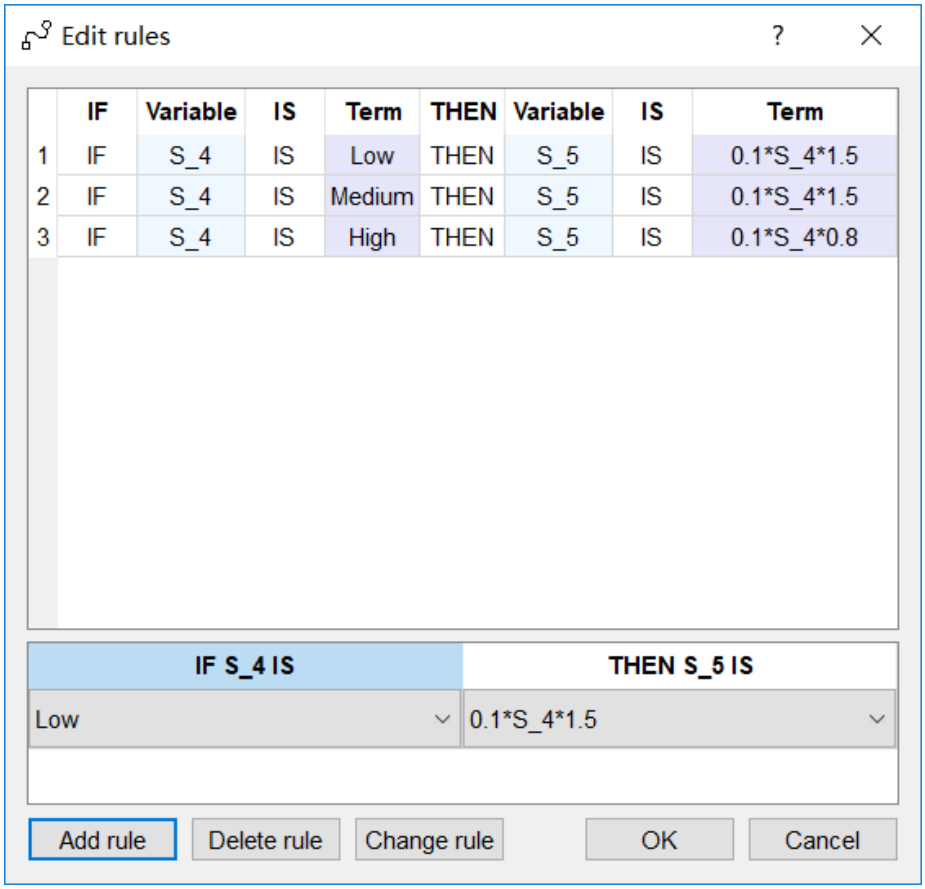


Figure 36: Rules of the 1D Diffusion Reaction model using the hybrid FIS setting (T-S).

4.1.3 Simulation result

In this example, we give the results of the four modes: no FIS (only ODEs), Mamdani, T-S, and hybrid FIS. The result of only ODEs is shown in Figure 37.

When using the Mamdani setting, we choose to assign an FIS to the arc from t_5_4 to S_4. The result is shown in Figure 38.

When using T-S, we choose to assign an FIS to the arc the arc from t_4_5 to S_5. The result is shown in Figure 39.

When using hybrid FIS, we choose to assign an mamdani FIS to the arc from t_5_4 to S_4 and to assign a T-S FIS to the arc from t_4_5 to S_5. The result is shown in Figure 40.

In the mode of only ODEs, the values of all variables eventually reach 2 while in the other three modes, the final values of the variables are not 2, but are very close to 2. And the trends of all four curves are consistent.

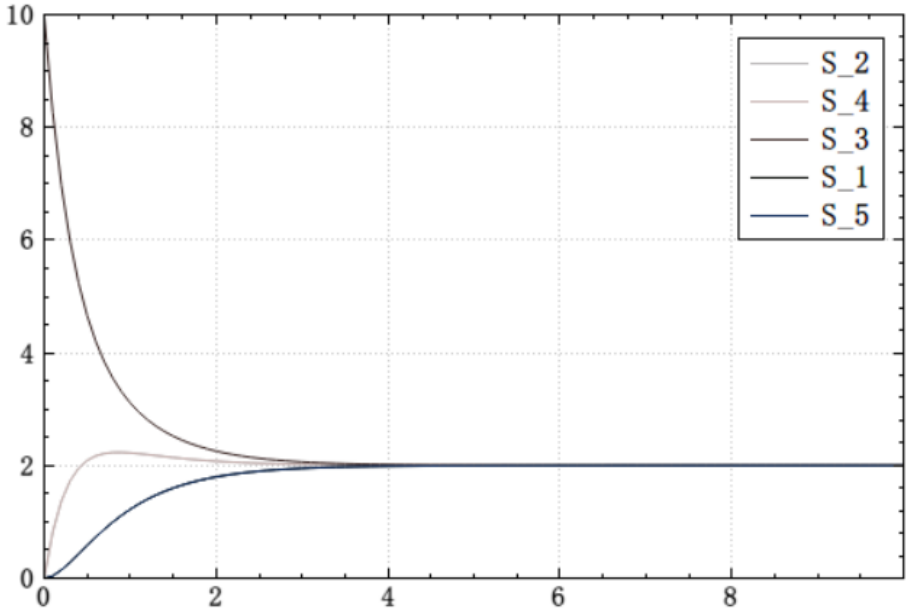


Figure 37: Simulation result of 1D diffusion reaction using only ODEs.

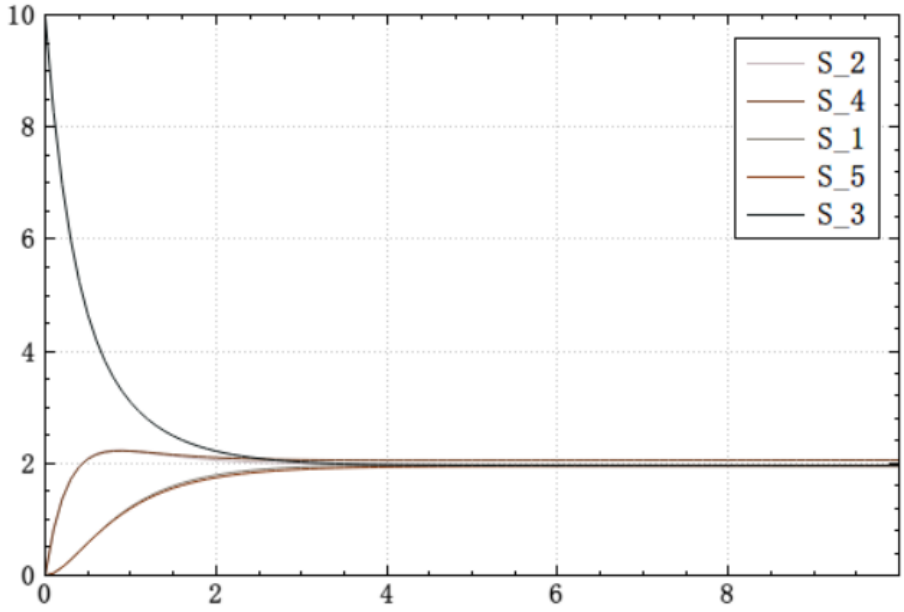


Figure 38: Simulation result of 1D diffusion reaction using Mamdani.

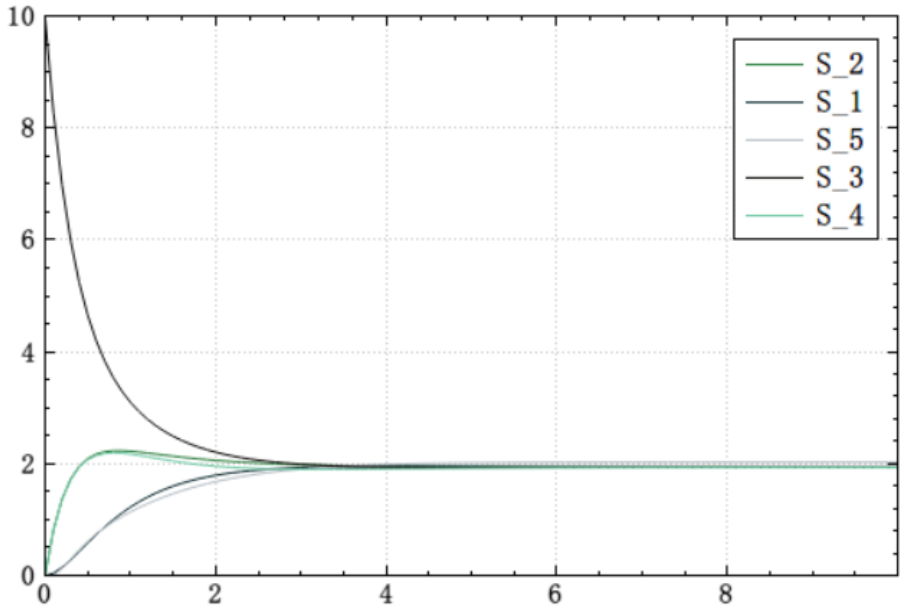


Figure 39: Simulation result of 1D diffusion reaction using T-S.

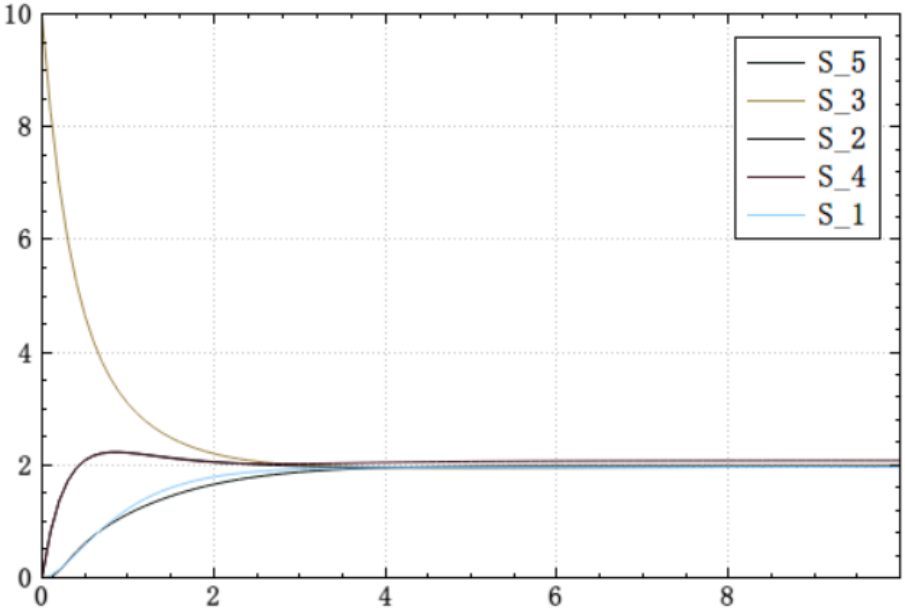


Figure 40: Simulation result of 1D diffusion reaction using hybrid FIS.

4.2 Enzymatic reaction

4.2.1 Introduction

The second example is enzymatic reaction [11]. Enzymes are macromolecular biological catalysts, which can accelerate chemical reactions. The molecules upon which enzymes may act are called substrates and the enzyme converts the substrates into different molecules, known as products. Almost all metabolic processes in a cell need enzyme catalysis in order to occur at rates fast enough to sustain life. Metabolic pathways depend upon enzymes to catalyze individual steps.

4.2.2 Model

The model is shown in Figure 41.

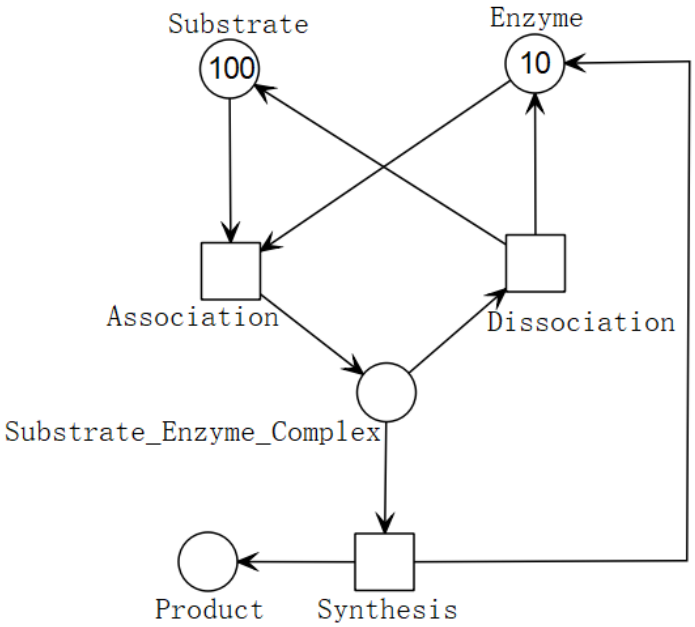


Figure 41: The model of enzymatic reaction.

Table 3: Transition functions of the enzymatic reaction model.

Transition	Function
Assoication	MassAction(0.1)
Dissociation	MassAction(0.1)
Synthesis	MassAction(0.1)

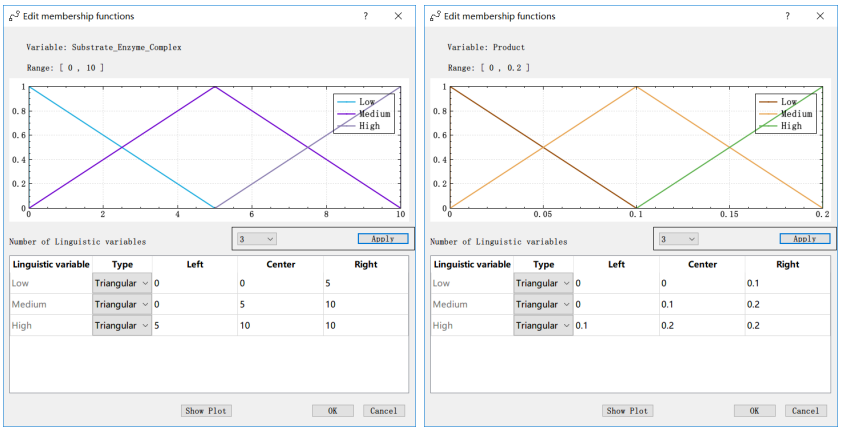


Figure 42: Membership functions of the enzymatic reaction model using the Mamdani setting.

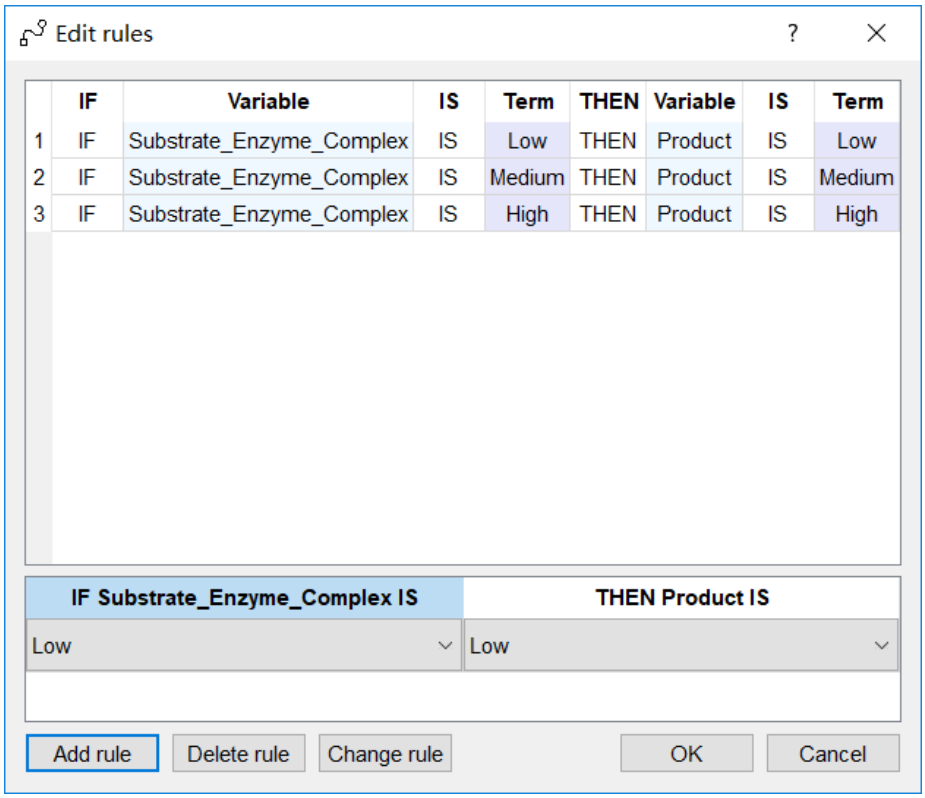


Figure 43: Rules of the enzymatic reaction model using the Mamdani setting.

4.2.3 Simulation result

In this example, we give the result of two modes: only ODEs and Mamdani. The result of only ODEs is shown in Figure 44.

When using the Mamdani setting, we choose to assign an FIS to the arc from Synthesis to Product. The result is shown in Figure 45.

As we can see, the results are almost the same.

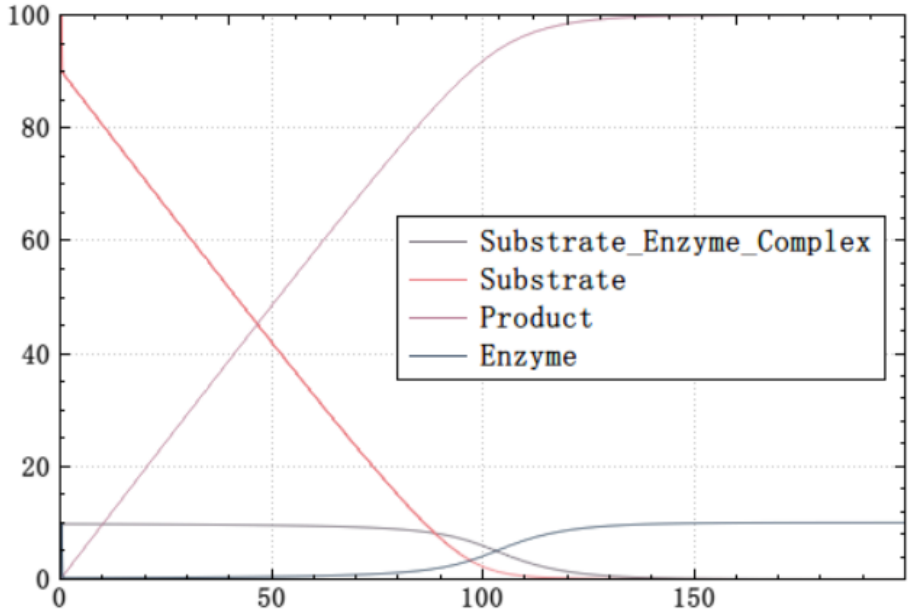


Figure 44: Simulation result of the enzymatic reaction model using only ODEs.

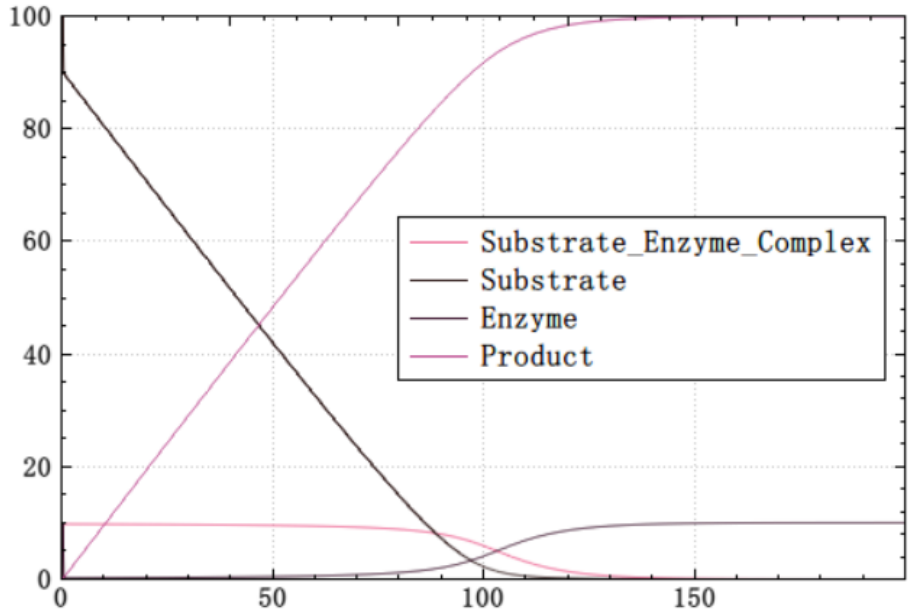


Figure 45: Simulation result of the enzymatic reaction model using the Mamdani setting.

4.3 RKIP pathway

4.3.1 Introduction

The Raf kinase inhibitor protein (RKIP) [12] is a kinase inhibitor protein that regulates many signaling pathways within the cell. RKIP is a member of the phosphatidylethanolamine-binding protein family and has displayed disruptive regulation on the Raf-1-MEK1/2, ERK1/2 and NF-kappaB signalling pathways, by interaction with the Raf-1 kinase.

4.3.2 Model

The model is shown in Figure 46, based on the CPN model given in [13].

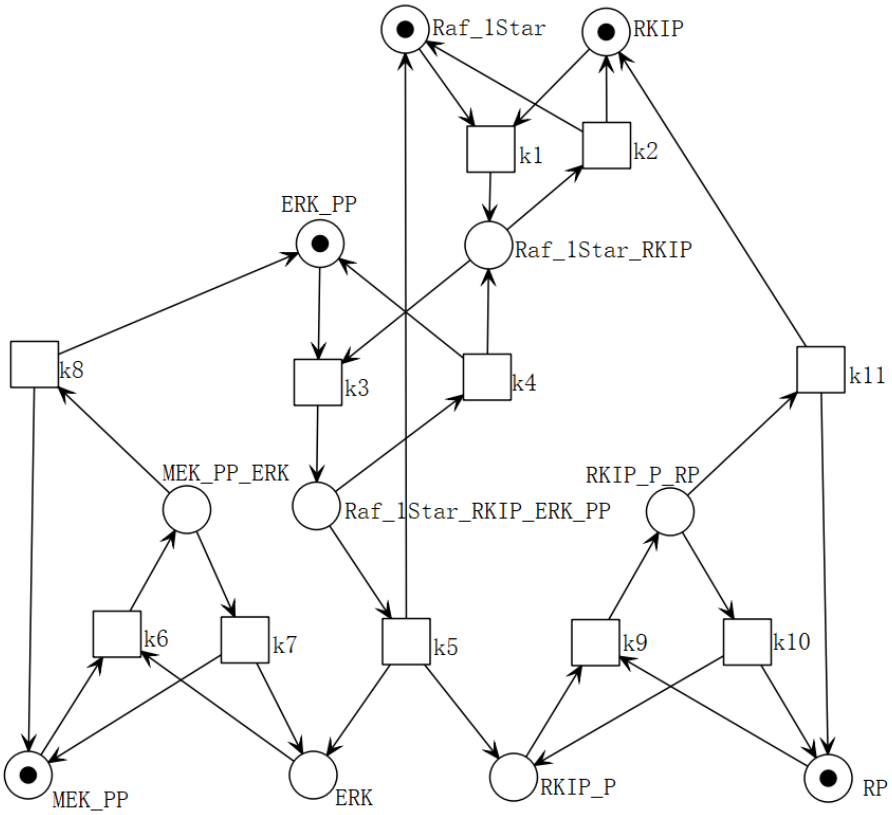


Figure 46: The model of the RKIP pathway.

4.3.3 Simulation result

In this example, we give the results of the modes: only ODEs and T-S. The result of only ODEs is shown in Figure 49.

When using T-S, we choose to assign an FIS to the arc from k3 to Raf_1Star_RKIP_ERK_PP. The result is shown in Figure 50.

Table 4: Transition functions of the RKIP pathway model.

Transition	Function
k1	MassAction(0.53)
k2	MassAction(0.0072)
k3	MassAction(0.625)
k4	MassAction(0.00245)
k5	MassAction(0.0315)
k6	MassAction(0.6)
k7	MassAction(0.0075)
k8	MassAction(0.071)
k9	MassAction(0.92)
k10	MassAction(0.00122)
k11	MassAction(0.87)

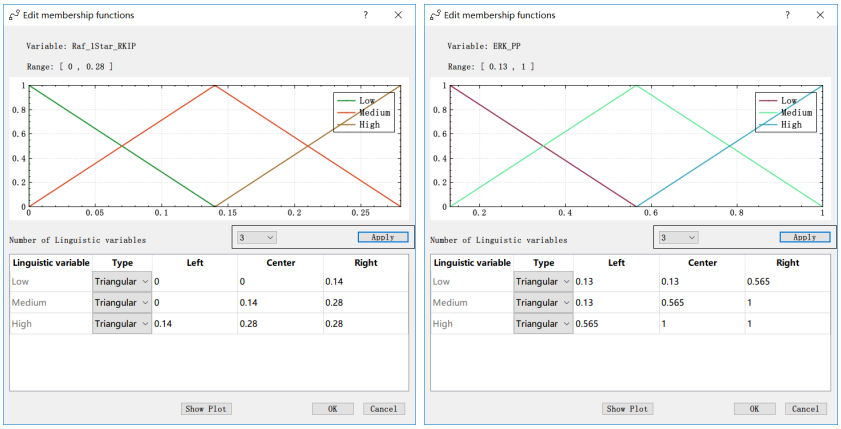
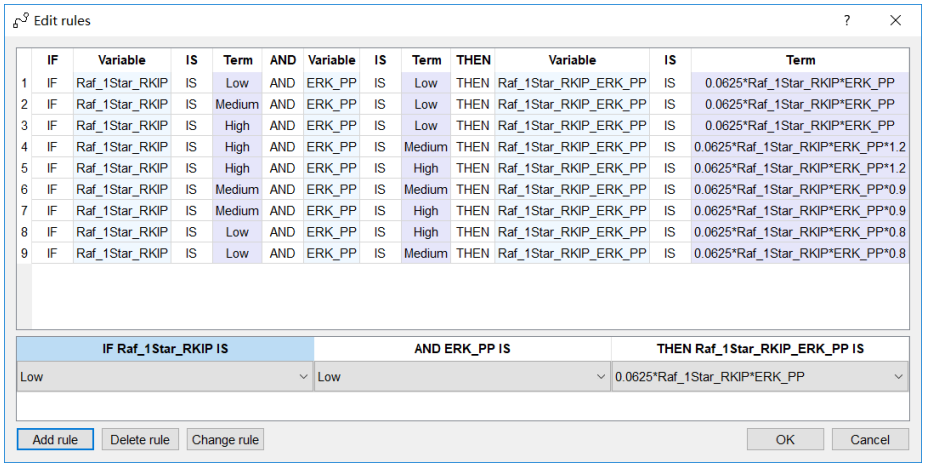


Figure 47: Membership functions of the RKIP pathway model using the T-S setting.

As we can see, the results are almost consistent. Pay attention to the bold blue curves of Figure 49 and Figure 50. The final result is a little different. But we can adjust the rules to get a better result.



	IF	Variable	IS	Term	AND	Variable	IS	Term	THEN	Variable	IS	Term
1	IF	Raf_1Star_RKIP	IS	Low	AND	ERK_PP	IS	Low	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP
2	IF	Raf_1Star_RKIP	IS	Medium	AND	ERK_PP	IS	Low	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP
3	IF	Raf_1Star_RKIP	IS	High	AND	ERK_PP	IS	Low	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP
4	IF	Raf_1Star_RKIP	IS	High	AND	ERK_PP	IS	Medium	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*1.2
5	IF	Raf_1Star_RKIP	IS	High	AND	ERK_PP	IS	High	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*1.2
6	IF	Raf_1Star_RKIP	IS	Medium	AND	ERK_PP	IS	Medium	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*0.9
7	IF	Raf_1Star_RKIP	IS	Medium	AND	ERK_PP	IS	High	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*0.9
8	IF	Raf_1Star_RKIP	IS	Low	AND	ERK_PP	IS	High	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*0.8
9	IF	Raf_1Star_RKIP	IS	Low	AND	ERK_PP	IS	Medium	THEN	Raf_1Star_RKIP_ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*0.8

Figure 48: Rules of the RKIP pathway model using the T-S setting.

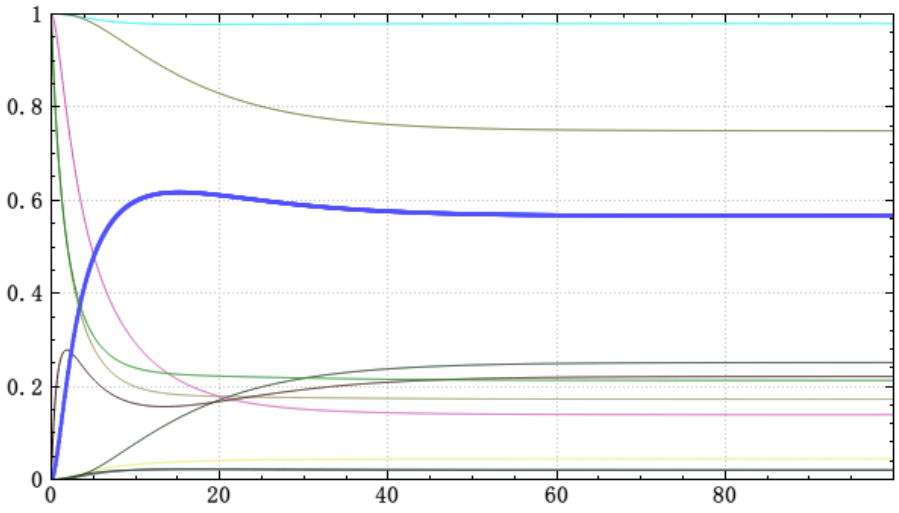


Figure 49: Simulation result of the RKIP pathway model using only ODEs.

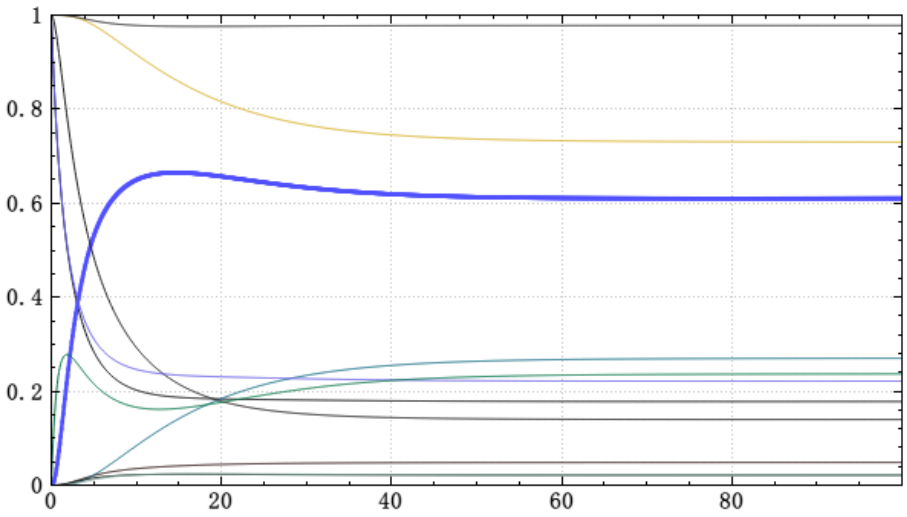


Figure 50: Simulation result of the RKIP pathway model using the T-S setting.

4.4 6-mercaptopurine metabolism

4.4.1 Introduction

6-Mercaptopurine (6-MP) is one of the important chemotherapy drugs used for treating acute lymphocytic leukaemia (ALL). According to the model given in [14, 15], we build the FCPN model of the 6-mercaptopurine metabolism.

4.4.2 Model

The model is shown in Figure 51.

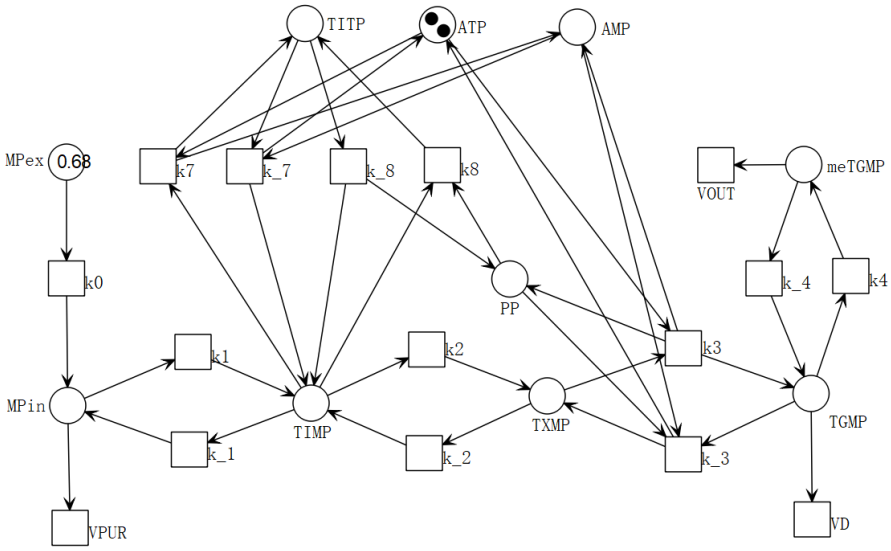


Figure 51: The model of 6-MP metabolism.

4.4.3 Simulation result

In this example, we give the results of three modes: only ODEs, Mamdani, and T-S. The result of only ODEs is shown in Figure 56.

When using Mamdani, we choose to assign an FIS to the arc from k0 to MPin. The result is shown in Figure 57.

When using T-S, we choose to assign an FIS to the arc from k0 to MPin. The result is shown in Figure 58.

As we can see, the results are almost the same.

Table 5: Transition functions of the 6-MP metabolism model.

Transition	Function
k0	MassAction(5)
k1	MassAction(10)
k_1	MassAction(0.01)
k2	MassAction(10)
k_2	MassAction(4)
k3	MassAction(5)
k_3	MassAction(0.01)
k4	MassAction(0.00001)
k_4	MassAction(0.1)
k7	MassAction(0.01)
k_7	MassAction(1)
k8	MassAction(0.5)
k_8	MassAction(0.01)
VPUR	MassAction(0.01)
VD	MassAction(0.9)
VOUT	MassAction(0.0001)

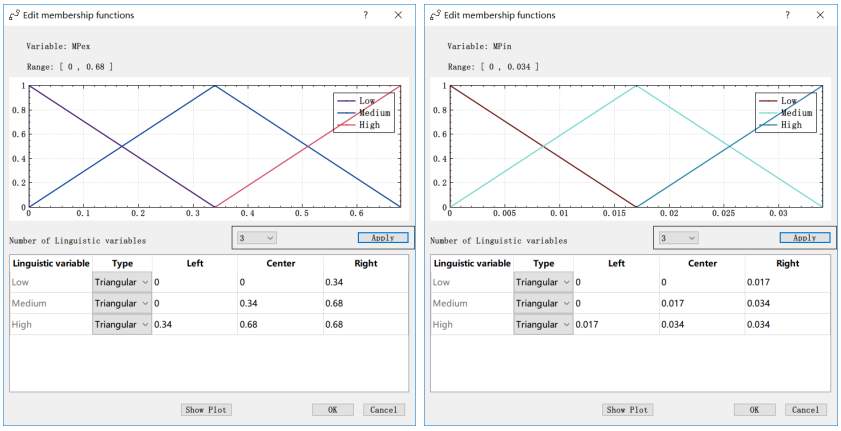


Figure 52: Membership functions of the 6-MP metabolism model using the Mamdani setting.

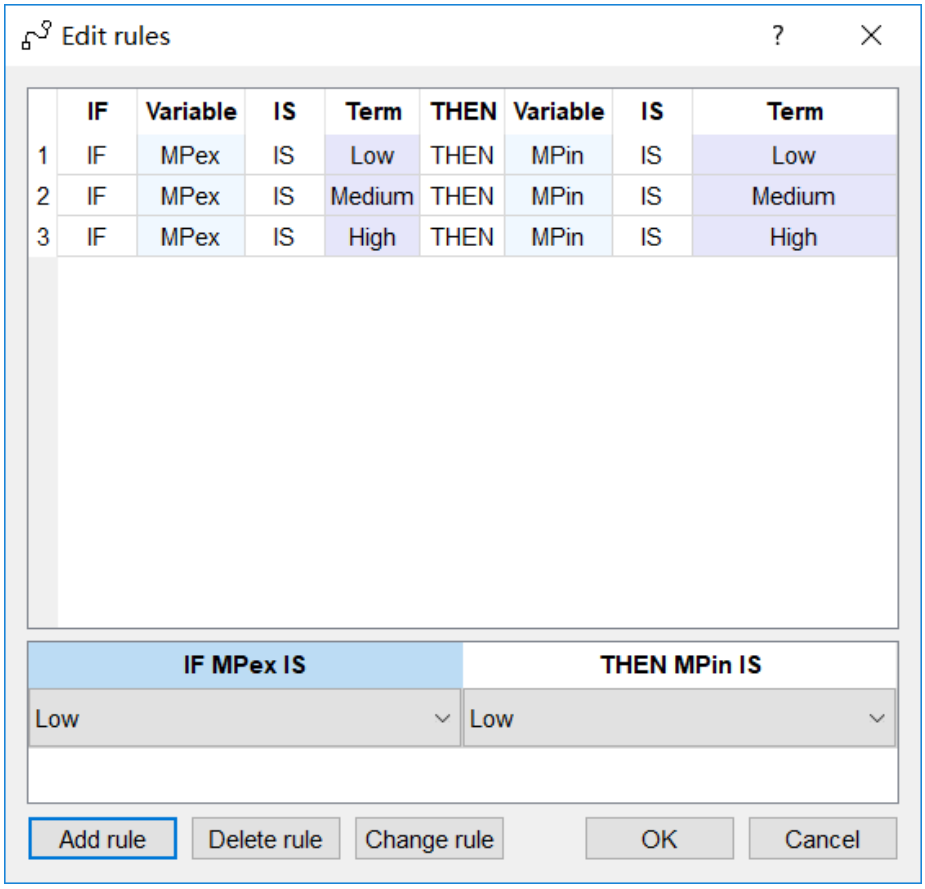


Figure 53: Rules of the 6-MP metabolism model using the Mamdani setting.

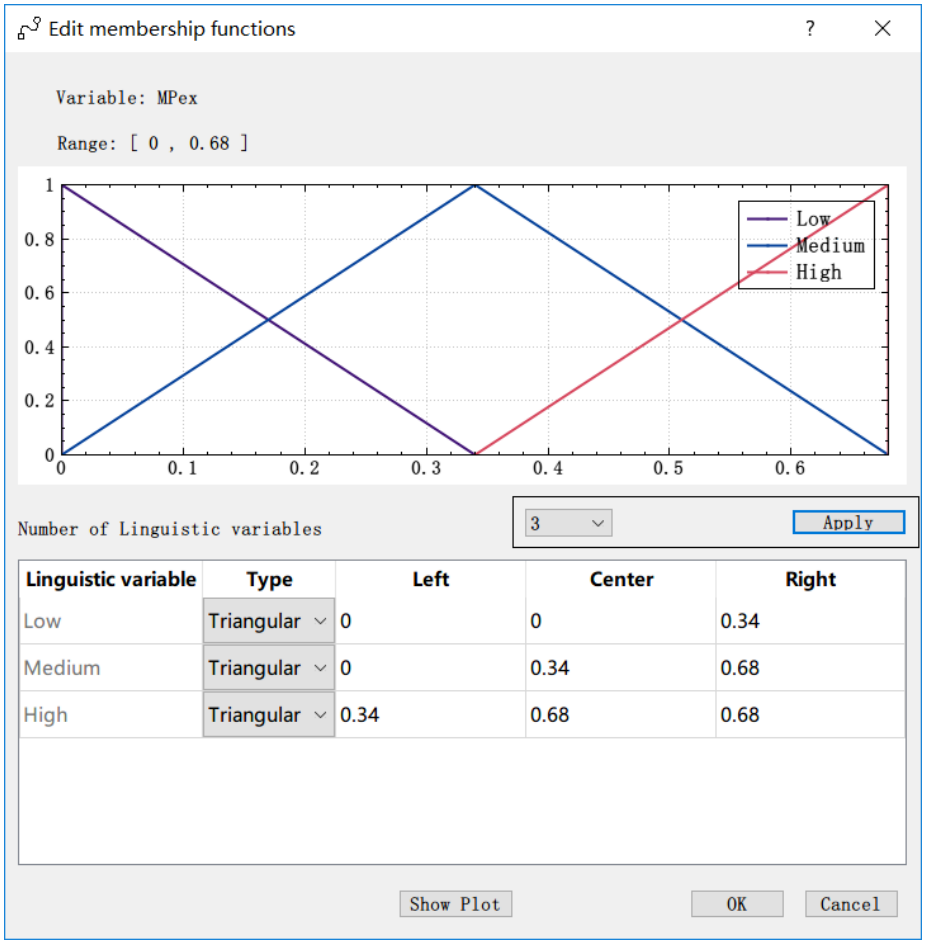


Figure 54: Membership functions of the 6-MP metabolism model using the T-S setting.

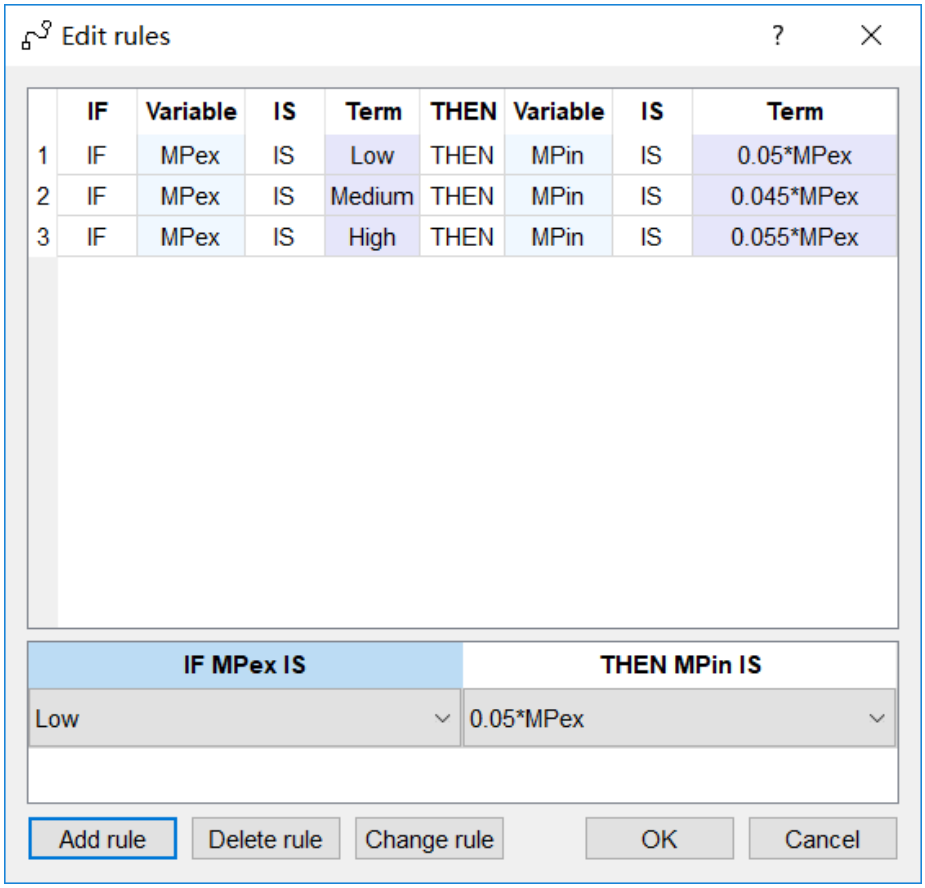


Figure 55: Rules of the 6-MP metabolism model using the T-S setting.

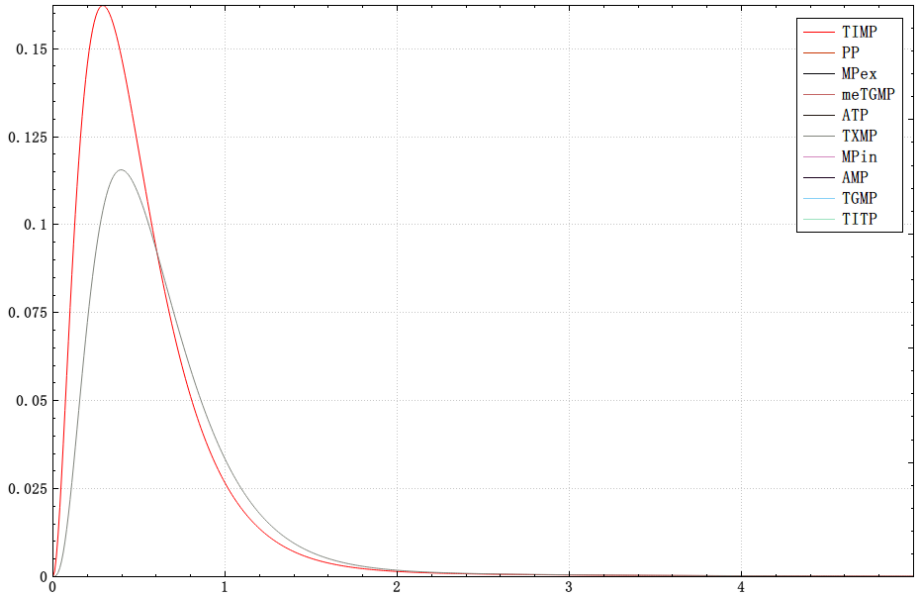


Figure 56: Simulation result of the 6-MP metabolism model using only ODEs.

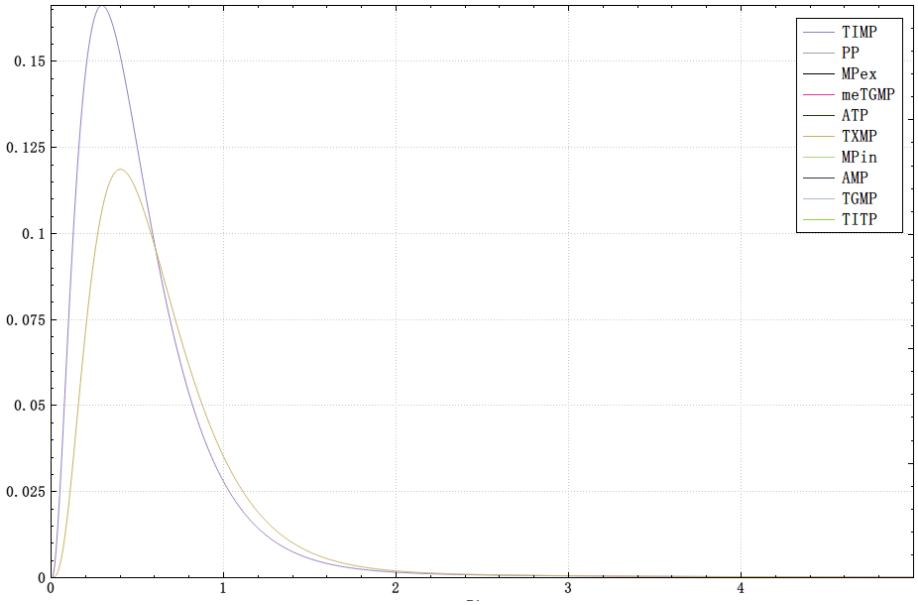


Figure 57: Simulation result of the 6-MP metabolism model using the Mamdani setting.

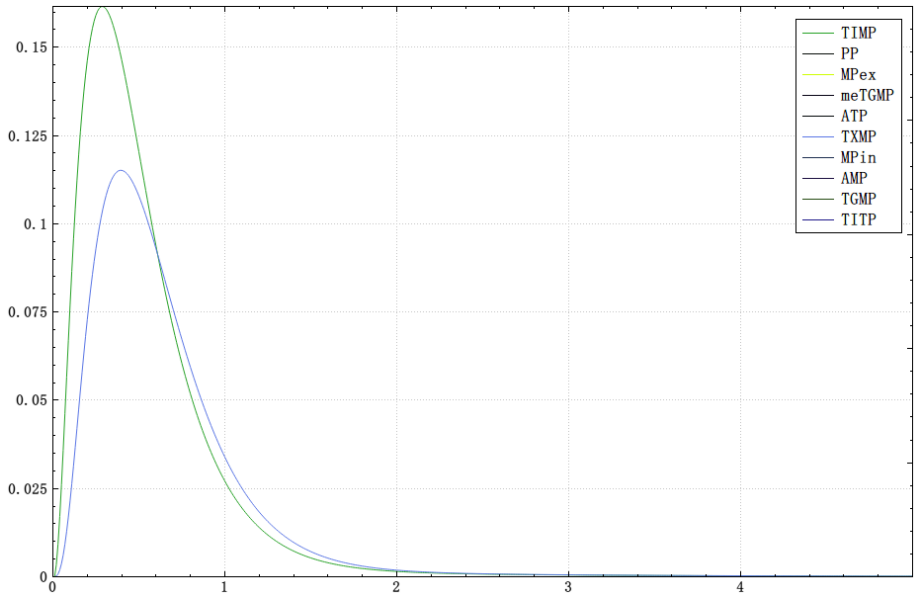


Figure 58: Simulation result of the 6-MP metabolism model using the T-S setting.

5 Combine FCPN with neuro-fuzzy system

5.1 Introduction

Since some parameters of fuzzy sets of Mamdani and T-S are not easy to determine, some experts have proposed a method of using neural networks to train datas to obtain fuzzy set parameters and rules [16][17].

In order to make it easier for users to train and get relevant parameters, we have added the function of automatically exporting .m files in the software. Users only need to open the file with MATLAB, fill in the relevant data, and run it to get the result.

5.2 Instruction

5.2.1 Output the file

If you want to output the .m file, you should follow the instruction in section 2.2. It does not matter if you are not sure about the correct parameters. However, make sure to fill in the appropriate rules which will be used in MATLAB. Of course, you can edit the rules in MATLAB as well. After that, you will get a result like Figure 59.

Left click the "Output to .m file" button to output the file.

5.2.2 Train datas in MATLAB

- **Mamdani**

  If the type of FIS is Mamdani, you will get two files. The comment content describes the variables to help you understand the code better.

  Part of the code in the file is similar to Figure 60. You should import the training datas to **input_train** and **out_train** using **csvread('data.csv')** or something like this. The **train_m** and **train_sigma** should also be editted by yourself. Make sure the orders of input datas should be consistent with **input_train**, **train_m**, **train_sigma**, and **rules**.

  rate, epoch(η), a, b, kk can also be editted to achieve better result. The following equation illustrates the role of a, b, and kk.

$$\eta = \begin{cases} a * \eta & \text{E(t+1)/E(t)} < 1 \\ b * \eta & \text{E(t+1)/E(t)} \geq \text{kk} \\ \eta & \text{other} \end{cases} \tag{1}$$

  where E(t+1) is the new error and E(t) is the old error.

  In the end, run the code to get the trained parameters and fill them in FCPN. Make sure the type of membership functions should be selected as **Gauss** and the AND Method should be selected as **Minimum**.

- **T-S**

  If the type of FIS is T-S, you will get one file. The comment content describes the variables to help you understand the code better.

  Part of the code in the file is similar to Figure 61.

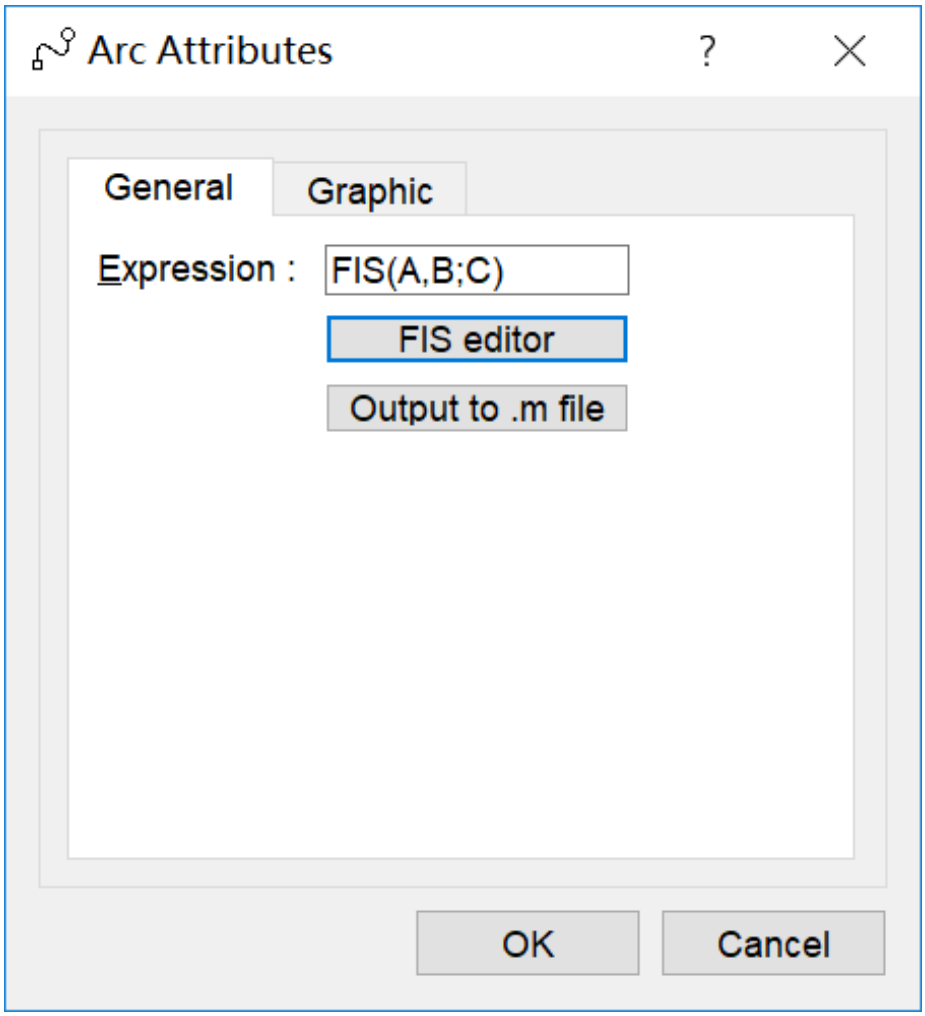


Figure 59: Output the file.

You should import the training datas to **trnData** using **csvread('data.csv')** or something like this. Make sure the output data should be in the last column.

You can also adjust the **epoch**.

In the end, run the code to get the trained parameters and fill them in FCPN. Make sure the type of membership functions should be selected as **Gauss** and the AND Method should be selected as **Product**.

```
%%%%%%%%%%Please enter the values of following variables%%%%%%%%%%
input_train = zeros(numVars, n);   %2*n
output_train = zeros(1, n);  %1*n
train_m = [
      0, 0, 0;
      0, 0, 0;
];
train_sigma = [
      0, 0, 0;
      0, 0, 0;
];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 60: Part of the code in Mamdani type file.

```
%%%%%%%%%%Please enter the data%%%%%%%%%%
trnData = zeros(n, numVars+1);    %n*2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 61: Part of the code in T-S type file.

References

[1] H Kitano. Systems biology: A brief overview. *Science*, 295(5560):1662–1664, 2002.

[2] JL Gouzé, A Rapaport, and MZ Hadj-Sadok. Interval observers for uncertain biological systems. *Ecological Modelling*, 133(1):45 – 56, 2000.

[3] M Heiner R Breitling, D Gilbert and R Orton. A structured approach for the engineering of biochemical network models, illustrated for signalling pathways. *Briefings in Bioinformatics*, 9(5):404–421, 2008.

[4] LA Zadeh. Fuzzy sets. *Inform. and Control*, 8(3):338–353, 1965.

[5] PP Wang, D Ruan, and EE Kerre. *Fuzzy Logic*. Springer-Verlag, Berlin Heidelberg, 2007.

[6] F Liu, M Heiner, and D Gilbert. Fuzzy Petri nets for modelling of uncertain biological systems. *Briefings in Bioinformatics*, 2018. accepted for publication:16 November 2018, published: 27 December 2018.

[7] T Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[8] E. H. Mamdani. Application of fuzzy algorithms for control of simple dynamic plant. *Proceedings of the Institution of Electrical Engineers*, 121(12):1585–1588, December 1974.

[9] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15(1):116–132, Jan 1985.

[10] F Liu, MA Blätke, M Heiner, and M Yang. Modelling and simulating reaction¨Cdiffusion systems using coloured Petri nets. *Computers in Biology and Medicine*, 53:297–308, October 2014. online July 2014.

[11] MA Blatke, M Heiner, and W Marwan. Tutorial - Petri Nets in Systems Biology. Technical report, Otto von Guericke University Magdeburg, Magdeburg Centre for Systems Biology, August 2011.

[12] Cho Kwang-Hyun, Shin Sung-Young, Kim Hyun-Woo, Olaf Wolkenhauer, Brian McFerran, and Walter Kolch. Mathematical modeling of the influence of rkip on the erk signaling pathway. In Corrado Priami, editor, *Computational Methods in Systems Biology*, pages 127–141, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[13] D. Gilbert and M. Heiner. From Petri nets to differential equations - an integrative approach for biochemical network analysis. In *Proc. ICATPN 2006*, pages 181–200. LNCS 4024, Springer, 2006.

[14] Anastasia I Lavrova, Eugene B Postnikov, Andrey Yu Zyubin, and Svetlana V Babak. Ordinary differential equations and boolean networks in application to modelling of 6-mercaptopurine metabolism. *Royal Society open science*, 4(4):160872, 2017.

[15] Anastasia I Lavrova, Eugene B Postnikov, Andrey Yu Zyubin, and Svetlana V Babak. Ode and random boolean networks in application to modelling of 6-mercaptopurine metabolism. *arXiv preprint arXiv:1611.00054*, 2016.

[16] J-SR Jang. Anfis: adaptive-network-based fuzzy inference system. *IEEE transactions on systems, man, and cybernetics*, 23(3):665–685, 1993.

[17] L. U. Xiao-Feng, Xiao Min Wang, and L. I. Guang-Yao. Research on phase-sensitive track circuit fault diagnosis based on mamdani fuzzy neural network. *Railway Standard Design*, 2015.