



Brandenburgische  
Technische Universität  
Cottbus - Senftenberg

Faculty 1  
Mathematics, Computer Science,  
Physics, Electrical Engineering and  
Information Technology

Institute of Computer Science

COMPUTER SCIENCE REPORTS

Report 01/17

March 2017

(Coloured) Hybrid Petri Nets in Snoopy –  
User Manual

Mostafa Herajy  
Fei Liu  
Christian Rohr  
Monika Heiner

Computer Science Reports  
Brandenburg University of Technology Cottbus - Senftenberg  
ISSN: 1437-7969

Send requests to: BTU Cottbus - Senftenberg  
Institut für Informatik  
Postfach 10 13 44  
D-03013 Cottbus

Mostafa Herajy  
Fei Liu  
Christian Rohr  
Monika Heiner  
snoopy@informatik.tu-cottbus.de  
<http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy>

## (Coloured) Hybrid Petri Nets in Snoopy – User Manual

Computer Science Reports  
01/17  
March 2017

Brandenburg University of Technology Cottbus - Senftenberg  
Faculty of Mathematics, Computer Science, Physics, Electrical Engineering and  
Information Technology  
Institute of Computer Science

Computer Science Reports  
Brandenburg University of Technology Cottbus - Senftenberg  
Institute of Computer Science

Head of Institute:

Prof. Dr. Hartmut König  
BTU Cottbus - Senftenberg  
Institut für Informatik  
Postfach 10 13 44  
D-03013 Cottbus

hartmut.koenig@b-tu.de

Research Groups:

Computer Engineering  
Computer Network and Communication Systems  
Data Structures and Software Dependability  
Database and Information Systems  
Programming Languages and Compiler Construction  
Software and Systems Engineering  
Theoretical Computer Science  
Graphics Systems  
Systems  
Distributed Systems and Operating Systems  
Internet-Technology

Headed by:

Prof. Dr. H. Th. Vierhaus  
Prof. Dr. H. König  
Prof. Dr. M. Heiner  
Prof. Dr. I. Schmitt  
Prof. Dr. P. Hofstedt  
Prof. Dr. C. Lewerentz  
Prof. Dr. K. Meer  
Prof. Dr. D. Cunningham  
Prof. Dr. R. Kraemer  
Prof. Dr. J. Nolte  
Prof. Dr. G. Wagner

CR Subject Classification (1998): D.2.2, D.2.7, I.6.5, I.6.8, J.3

Printing and Binding: BTU Cottbus - Senftenberg

ISSN: 1437-7969

# **(Coloured) Hybrid Petri Nets in Snoopy – User Manual**

---

<http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy>

8<sup>th</sup> March, 2017

**Mostafa Herajy, Fei Liu, Christian Rohr, and Monika Heiner**  
snoopy@informatik.tu-cottbus.de

<http://www-dssz.informatik.tu-cottbus.de>

Data Structures and Software Dependability

Computer Science Department

Brandenburg University of Technology Cottbus-Senftenberg

## Contents

1	Introduction	1
2	Preliminaries	2
3	Installation	2
3.1	Downloading Snoopy . . . . .	2
3.2	Installing Snoopy . . . . .	3
3.2.1	MS Windows . . . . .	3
3.2.2	Mac OSX . . . . .	3
3.2.3	Linux (Ubuntu) . . . . .	4
4	Getting Started	5
4.1	The Water Tank Model . . . . .	5
4.2	T7 Phage Model . . . . .	6
5	Constructing Hybrid Models	7
5.1	Frequent Tasks . . . . .	8
5.1.1	Creating a New Net . . . . .	9
5.1.2	Saving and Loading . . . . .	10
5.2	Hybrid Petri Nets . . . . .	12
5.2.1	Drawing a Net . . . . .	12
5.3	$HPN$ elements in Snoopy . . . . .	12
5.3.1	Places . . . . .	13
5.3.2	Transitions . . . . .	14
5.3.3	Arcs . . . . .	16
5.3.4	Marking-dependent Arc Weights . . . . .	18
5.3.5	Meta Data . . . . .	18
5.3.6	Connection Rules . . . . .	19
5.4	Moving, Editing and Deleting Elements . . . . .	20
5.5	Adjusting Element Properties . . . . .	21
5.5.1	Name . . . . .	21
5.5.2	Marking . . . . .	21
5.5.3	Show/Hide Attribute . . . . .	22

5.5.4	Continuous Rates . . . . .	23
5.5.5	Stochastic Rates . . . . .	24
5.5.6	Delay . . . . .	25
5.5.7	Weights . . . . .	26
5.5.8	Graphics . . . . .	27
5.5.9	Arc Style . . . . .	28
5.5.10	Other Properties . . . . .	29
5.5.11	Converting Between Elements . . . . .	29
5.6	Constants and Functions . . . . .	30
5.6.1	Defining a New Constant . . . . .	31
5.6.2	Defining a New Constant Group . . . . .	32
5.6.3	Deleting a Constant . . . . .	33
5.6.4	Manipulating Constant Value Sets . . . . .	33
5.6.5	Using Constants in Element Properties . . . . .	33
5.7	Coloured Hybrid Petri Nets . . . . .	33
5.7.1	Defining Colour Sets . . . . .	34
5.7.2	Defining Constants . . . . .	38
5.7.3	Defining Variables . . . . .	38
5.7.4	Annotating Elements with Colours . . . . .	39
6	Model Simulation . . . . .	42
6.1	Launching the Simulation Dialog . . . . .	42
6.2	Starting the Simulation . . . . .	44
6.3	Model Configuration . . . . .	45
6.3.1	Adjusting the Initial Marking . . . . .	45
6.3.2	Adjusting Transition Rates . . . . .	45
6.3.3	Adjusting Parameters . . . . .	46
6.4	Simulation Configuration . . . . .	46
6.4.1	Simulation Intervals . . . . .	47
6.4.2	ODE Solver Settings . . . . .	48
6.4.3	Time Synchronisation Setting . . . . .	51
6.5	Viewing Simulation Results . . . . .	52
6.5.1	Opening the default view . . . . .	53
6.5.2	Manipulating Views . . . . .	54

- 6.5.3 Exporting Views . . . . . 55
- 6.5.4 Adding New Views . . . . . 57
- 6.5.5 Removing Existing Views . . . . . 59
- 6.5.6 Edit Viewer’s Properties . . . . . 59
- 7 Export and Import . . . . . 62
  - 7.1 Export to Other Petri Net Classes . . . . . 62
  - 7.2 Export to (C)ANDL File . . . . . 63
  - 7.3 Export as a Latex Report . . . . . 64
  - 7.4 Export to Other File Formats . . . . . 64
    - 7.4.1 Export to ODE . . . . . 65
  - 7.5 Import . . . . . 66
- 8 Useful resources . . . . . 67
- 9 Frequently Asked Questions . . . . . 68
- References . . . . . 71



## List of Figures

1	Snoopy Setup under Windows. . . . .	4
2	Snoopy Setup under Mac OSX. . . . .	4
3	Water Tank $\mathcal{HPN}$ Model . . . . .	6
4	Screenshot for the Simulation of the Water Tank Model in Snoopy . . . . .	7
5	T7 Phage $\mathcal{HPN}$ Model . . . . .	8
6	Screenshot for the Simulation of the T7 Phage Model in Snoopy . . . . .	8
7	Graphical User Interface under MS Windows . . . . .	9
8	Opening a New Petri Net Class . . . . .	10
9	A New $\mathcal{GHPN}$ File . . . . .	11
10	Save as Dialog . . . . .	11
11	Open Dialog . . . . .	12
12	Elements of $\mathcal{HPN}$ in Snoopy . . . . .	13
13	Petri Net Representation of a Simple Reaction . . . . .	17
14	Example with Marking-dependent Arc Weights . . . . .	19
15	Valid Connections between $\mathcal{HPN}$ 's Elements . . . . .	20
16	Changing the Place Name . . . . .	22
17	Changing the Place Marking . . . . .	23
18	Editing Transition Function. . . . .	25
19	Function Assistant. . . . .	26
20	Editing Immediate Transition Weight. . . . .	27
21	Example of Immediate Transition Weights . . . . .	28
22	Comparing the Different Edge Styles . . . . .	29
23	Changing the Setting of the Edge Style . . . . .	29
24	Screenshot of the <i>Convert to</i> Dialog . . . . .	30
25	Constant Dialog Box . . . . .	31
26	Adding a New Constant . . . . .	32
27	$\mathcal{HPN}$ Model of the Repressilator . . . . .	35
28	$\mathcal{HPN}^C$ Model of the Repressilator . . . . .	35
29	Coloured Declaration Panel . . . . .	36
30	Simple Colour Set Definition Dialog . . . . .	37
31	Compound Colour Set Definition Dialog . . . . .	37
32	Constant Definition Dialog for Coloured Petri Nets . . . . .	38

33	Variable Definition Dialog . . . . .	39
34	Place Property Dialog . . . . .	40
35	Transition Property Dialog . . . . .	41
36	Arc Property Dialog . . . . .	42
37	Screenshot of the Unfolding Dialog . . . . .	43
38	Screenshot of the Simulation Dialog . . . . .	44
39	Screenshot of Configuring a Model from the Simulation Dialog . . . . .	47
40	Simulation Options . . . . .	48
41	Different Settings of the ODE Solver . . . . .	50
42	Options for Customising the Hybrid Simulator . . . . .	53
43	Opening the Default View for the First Time . . . . .	54
44	Edit Node List Dialog . . . . .	55
45	Default View After Selecting Two Places . . . . .	56
46	Example of Tabular Viewer . . . . .	57
47	Example of Histogram Plot . . . . .	58
48	Export Properties Dialog . . . . .	58
49	Creating a New View . . . . .	59
50	Adjusting the Viewer Properties . . . . .	60
51	Export Dialog Box . . . . .	63
52	Export Properties Dialog Box for $(C)ANDL$ Files . . . . .	64
53	Export Properties Dialog Box for Latex Reports . . . . .	65
54	Simulation Dialog with Import/Export Details . . . . .	66
55	Example of Exporting an $HPN$ Model into a Set of ODEs . . . . .	67
56	Import Dialog . . . . .	68

## List of Tables

1	Shortcut keys of the different $\mathcal{HPN}$ elements in Snoopy . . . . .	15
2	Comparison of the different transition types in Snoopy $\mathcal{HPN}$ . . . . .	18

## 1 Introduction

Hybrid simulation of biological processes becomes widely used to overcome the limitations of the pure stochastic or the complete deterministic simulation. In this manual, we present easy-to-follow steps for constructing and executing hybrid models via Snoopy [HHL<sup>+</sup>12]. Snoopy is a tool to design and animate or simulate hierarchical graphs, i.e., qualitative, stochastic, continuous, and hybrid Petri nets. This manual is concerned with hybrid Petri nets ( $\mathcal{HPN}$ ) [HH12] as well as their coloured counterpart ( $\mathcal{HPN}^c$ ) [HLR14].  $\mathcal{HPN}$  combine the merits of stochastic and continuous Petri nets into one single class. Moreover,  $\mathcal{HPN}$  in Snoopy supports state of the art hybrid simulation algorithms (e.g., [HH16]) to execute the constructed  $\mathcal{HPN}$  models. Simulating a model using Snoopy's hybrid simulation involves first constructing the reaction network via  $\mathcal{HPN}$  notations and afterwards executing such model.

$\mathcal{HPN}$  in Snoopy entail two types of places: discrete and continuous. Discrete places can hold non-negative integer numbers indicating their tokens, while continuous places can hold non-negative real values as markings. In the biological context, the former place type can be used to represent biological species at the molecules level, while the latter one represents such species at the concentration level.

Furthermore, Snoopy  $\mathcal{HPN}$  offer a wide range of transitions: continuous, stochastic, deterministic, scheduled, and immediate. In addition,  $\mathcal{HPN}$  places are connected with transitions through a set of arcs: standard, read, inhibitor, equal, reset, and modifier arcs. The interconnections between places and transitions using these arcs are governed using some rules (see Section 5.3.6).

We also extend  $\mathcal{HPN}$  to support colours. Coloured hybrid Petri nets ( $\mathcal{HPN}^c$ ) [HLR14] combine  $\mathcal{HPN}$  with coloured Petri nets, which facilitates the modelling of large biological systems. When we use  $\mathcal{HPN}^c$  in Snoopy, we usually follow two main steps. We first construct a  $\mathcal{HPN}^c$  model for a biological system to be studied with the elements of both  $\mathcal{HPN}$  and coloured Petri nets. When the model is ready, we can automatically unfold it to an  $\mathcal{HPN}$ , and then run simulation on the  $\mathcal{HPN}$  automatically unfolded in the background.

Snoopy provides five different types of algorithms that can be used to simulate constructed  $\mathcal{HPN}$  models, including many state of the art approaches. In addition, the same model can also be simulated as stochastic, continuous, or a hybrid one despite their graphical representation. Such feature makes Snoopy  $\mathcal{HPN}$  more flexible to experiments with different kind of simulators. Beside, a coarse-grained shared memory parallelisation for

executing different concurrent runs is also supported such that the simulator can take advantage of the multi-core facilities available at nowadays computers.

Generally speaking, the organisation of this manual is divided into two main parts: modelling and simulation. In the modelling part, we present how the reader can use  $\mathcal{HPN}$  elements to model certain biological reactions, while in the simulation section, we focus on configuring, setting and executing the  $\mathcal{HPN}$  models.

We also discuss two methods of simulating the  $\mathcal{HPN}$  models: static and dynamic. In static partitioning, the model is simulated as it has been originally constructed by the user. In dynamic partitioning, the simulator engine decides on the fly which transitions are considered as continuous and which ones as stochastic based on some partitioning criteria.

A brief remark for readers who are not familiar with Petri net notations to model biochemical reactions: Snoopy adopts continuous Petri nets to represent a system of ordinary differential equations. The ODEs are generated automatically and can be exported to different formats. On the other side, stochastic Petri nets are simulated using the Gillespie direct method.

## 2 Preliminaries

This manual assumes that the reader already has a basic understanding of Petri nets including stochastic and continuous ones. However, no knowledge about hybrid Petri nets is required. Readers who are not familiar with such kinds of Petri nets, can find useful material at [BHM15]. Users who are already familiar with Snoopy can skip Section 3 and begin directly to read about constructing hybrid models.

## 3 Installation

This section presents a brief overview of installing Snoopy on three of the well-known operating systems.

### 3.1 Downloading Snoopy

Snoopy is a free and platform-independent Software. You can download it from its official website: <http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy>.

Therefore you can also find materials and examples about how you can use Snoopy. Snoopy can work under Windows, Mac and and for some Linux distributions.

## **3.2 Installing Snoopy**

The specific installation procedure depends on your operating system version. Below we give examples using Windows, Mac OS X, and Ubuntu.

### **3.2.1 MS Windows**

To install Snoopy under Windows, follow these steps:

1. Obtain the windows installer package for Snoopy. This should be an msi file.
2. Double click the obtained setup package.
3. The windows installer will start as shown in Figure 1.
4. Follow the simple instructions in this window by hitting the next button.
5. At the end of these steps Snoopy should be installed on your computer and a shortcut will be created at your desktop.
6. To start Snoopy, simply double click the Snoopy's shortcut from your desktop.

### **3.2.2 Mac OSX**

To install Snoopy under Mac OS X, first acquire a Snoopy version. The Snoopy setup file under Mac OS is in a disk image format (dmg). All the necessary data are provided in a single file. To install this file on your computer do the following steps:

1. Mount the disk image on your computer by double-clicking the dmg file.
2. The disk image will appear as another CD in your Finder with the name "Snoopy".
3. Opening this disk, you will find the Snoopy application bundle, as shown in Figure 2.
4. Copy Snoopy to your desired location, or drag and drop it into the application folder.

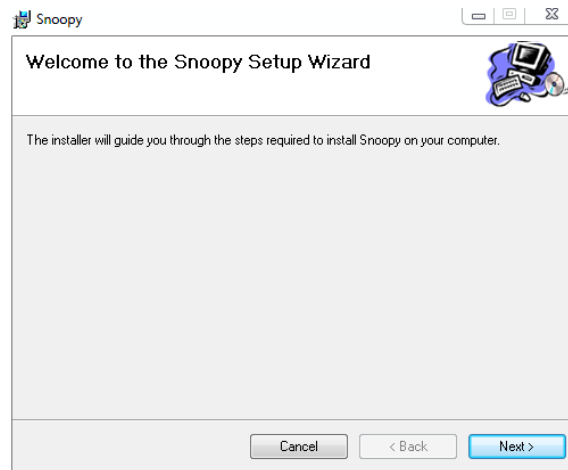


Figure 1: Snoopy Setup under Windows.



Figure 2: Snoopy Setup under Mac OSX.

### 3.2.3 Linux (Ubuntu)

To install and use Snoopy under Linux, follow these steps:

1. Download the Snoopy file `snoopy-stable-linux-ubuntu64-xx.tgz`
2. Extract the archive with `"tar xzfv snoopy-stable-linux*.tgz"`
3. Change into the directory `"cd snoopy2/bin"`
4. Start Snoopy by calling `"./snoopy2.sh"`

## 4 Getting Started

Before going into details of how to design and simulate  $\mathcal{HPN}$ , we present two simple examples and we show how to simulate them using Snoopy. These two examples are already implemented and you can find the corresponding Snoopy files at the accompanied subfolder called HPNs.

### 4.1 The Water Tank Model

We start with a simple and very popular example: the water tank; see, e.g., [Her13, BHM15]. Consider a water tank in which the water bumping is switched on/off based on the amount of water currently available in the tank. When the water level exceeds a certain amount, the bumping is switched off and the water level starts to decrease until a minimum amount is reached causing the water bumping to be switched on and the water starts to increase again.

To model this behaviour, we need to mix continuous and discrete elements. The switch on/off is modelled using discrete elements while the water tank is represented using a continuous element. Figure 3a gives an  $\mathcal{HPN}$  model of a water tank.

In this model the continuous place *water\_tank* is used to represent the amount of water at any time, since the amount of water is a continuous quantity. Two discrete places *On* and *Off* represent the discrete part of the system state at any time. The system states are changed through the two deterministic transitions *Off\_to\_On* and *On\_to\_Off*. The minimum and the maximum water level in the tank is 0.1 and 2.0 litre respectively. Inhibitor and read arcs are used to check if the water level is below or above the predefined level, respectively. Moreover, we used a scheduled transition to give a pulse to the amount of water at time 50 which is reflected in the simulation result in Figure 3b.

To try this model, follow these steps:

1. Open Snoopy on your computer.
2. From *file*, select *open* (see below for the detailed procedure).
3. Locate the Snoopy file: *water\_tank.hpn* at your system, which comes with this tutorial.
4. Hit *Open*.



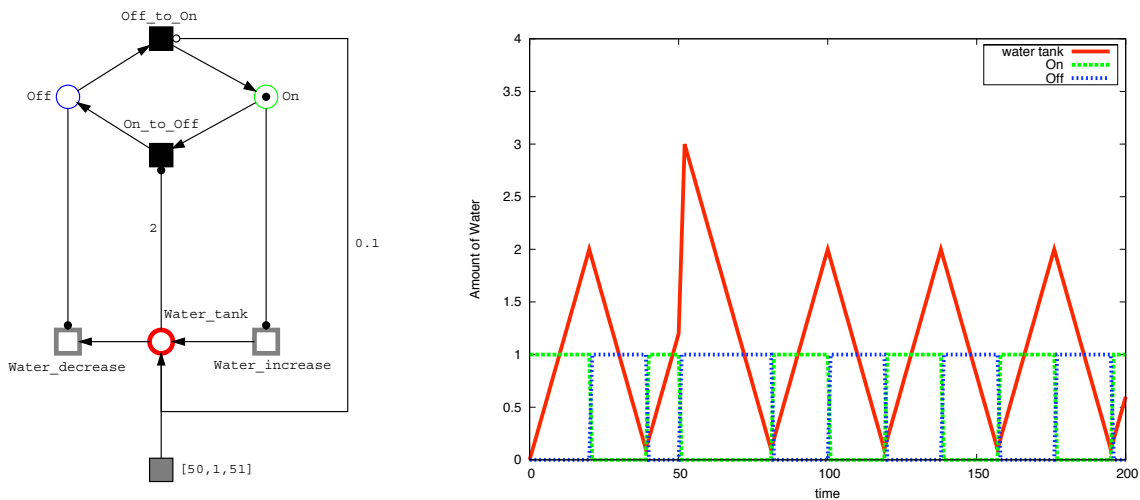


Figure 3: A simple example of constructing and simulating hybrid models using Snoopy HPN: (a) HPN representation of the water tank, and (b) simulation result.

5. From the *View* menu, select *Start Simulation-Mode* (See below for a detailed procedure).
6. The simulation dialog will open.
7. Click on the *Start* button.
8. Double click on the *on\_off* view to see the result (See below for a detailed procedure).
9. The simulation result will be as shown in Figure 4.

## 4.2 T7 Phage Model

Next we consider a simple model from the biological context; Figure 5a lists the set of reactions, while Figure 5b gives the HPN representation. The net is partitioned into discrete and continuous parts based on the reaction kinetic in Figure 5a. Here the reactions *R5* and *R6* are considered to be fast compared to the other four reactions. Therefore, they have been represented by two continuous transitions.

To try this model, follow these steps:

1. Locate the file *T7model* coming with this manual.

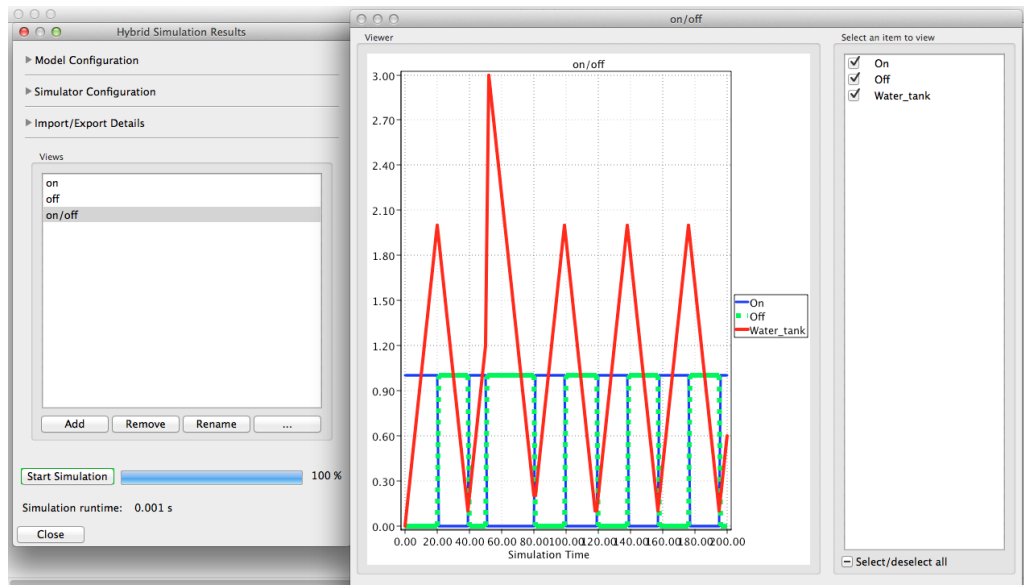


Figure 4: A screenshot for the simulation of the water tank model in Snoopy.

2. From *file*, select *open*.
3. Select the file *T7model.hpn*, then click *open*.
4. The Snoopy file will be opened.
5. From the *View* menu, select *Start Simulation-Mode* (See below for a detailed procedure), as we did in the previous example.
6. The simulation dialog will open.
7. Click on the *Start* button.
8. You will get the model simulated as shown in Figure 6.

## 5 Constructing Hybrid Models

In this section we present a stepwise procedure to show how an *HPN* model can be constructed via Snoopy. Later in Section 6, we discuss the simulation aspects.

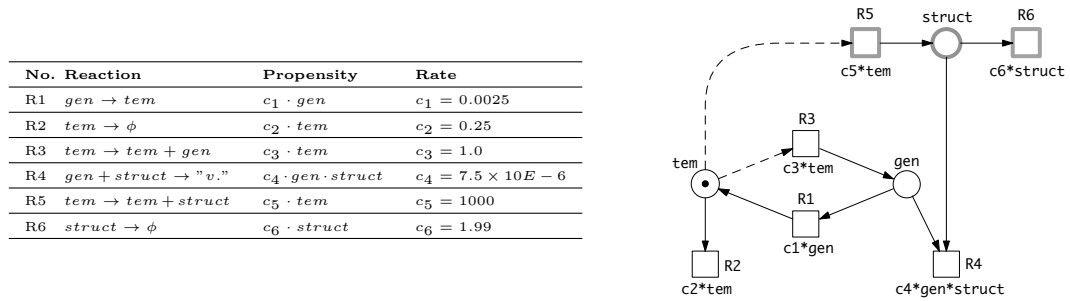


Figure 5: T7 Phage – an example of a simple biological network modelled using *HPN*: (a) the set of reactions according to [SYSY02], and (b) the corresponding *HPN* adopted from [HH12].

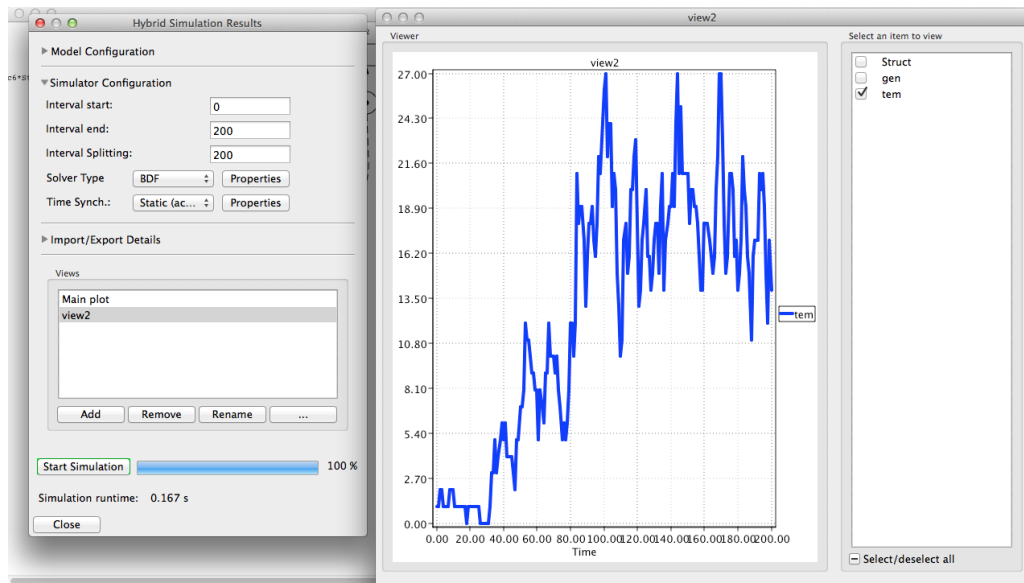


Figure 6: A screenshot for the simulation of the T7 Phage Model in Snoopy.

## 5.1 Frequent Tasks

There are a number of tasks that you will repeat frequently. Therefore, you will need to master them. These include: opening and closing Snoopy files, creating a new file, etc.



Figure 7: Graphical User Interface under MS Windows.

### 5.1.1 Creating a New Net

After installing Snoopy, you can open its GUI similar to other applications running on your local system. If you have trouble in opening it, please check the FAQ Section at the end of this tutorial. The user interface of Snoopy is shown in Figure 7 for the windows operating system.

To create a new *HPN* file, follow these steps:

- Select from *file* menu  $\rightarrow$  *new*.
- The *new class* dialog will open, as shown in Figure 8.
- Select *Hybrid Petri Net* from the class template list.
- Click on *Ok* button, a new Snoopy file will open with hybrid Petri nets elements, see Figure 9.

Now you can draw your net using the elements in the left part of Snoopy. The upcoming sections provide a detailed discussion of how to model and simulate a biochemical network using these elements.

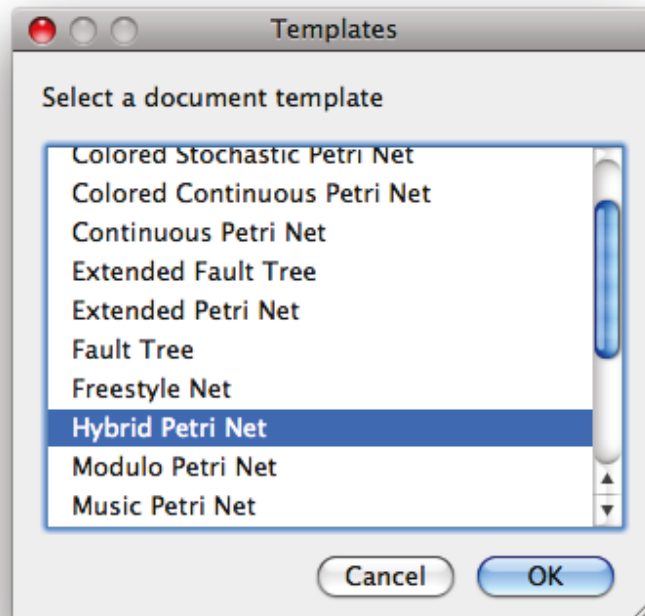


Figure 8: Opening a new Petri net class.

### 5.1.2 Saving and Loading

A frequent task that you will need to master when interacting with Snoopy is saving and loading a  $HPN$  model. To save your model for the first time follow these steps:

1. From *file* menu, select *save*.
2. The *save as* dialog box will open as in Figure 10.
3. Write a valid file name and click *save*.

This dialog will open when you save your file for the first time. A subsequent save will overwrite the previous file version. If you would like to give the file a different name, then select *file*  $\rightarrow$  *Save as*. Please remember to frequently save your file while editing your model so that you do not lose your work in case of sudden crashes. You can also use the popular  $Ctrl+S$  command to quickly save your Snoopy file.

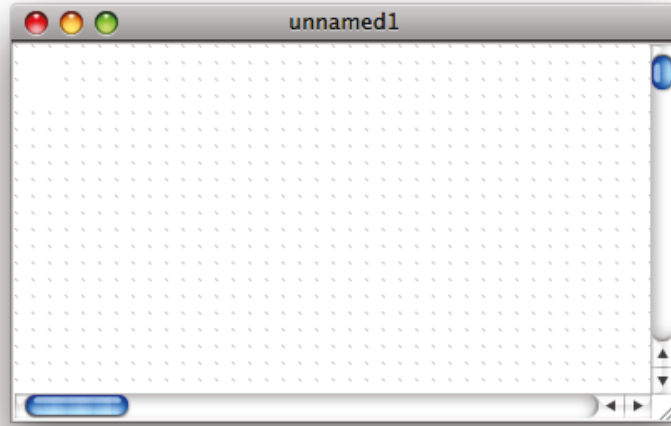


Figure 9: A new *GHPN* file.

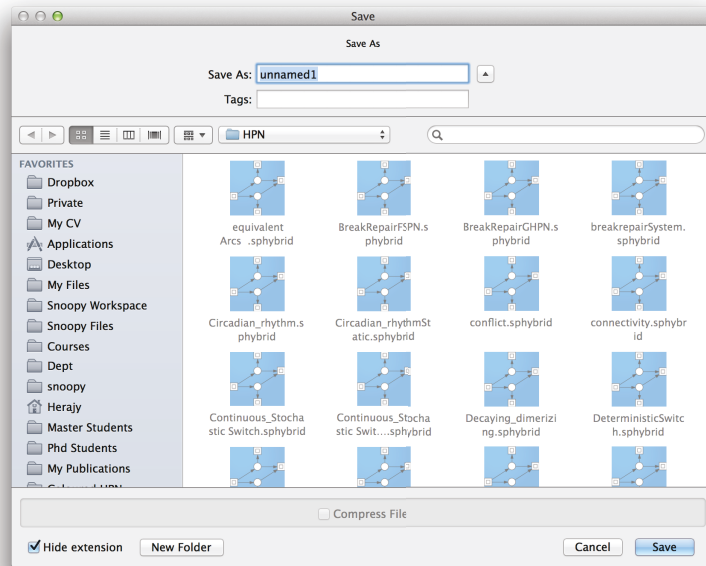


Figure 10: Save as dialog.

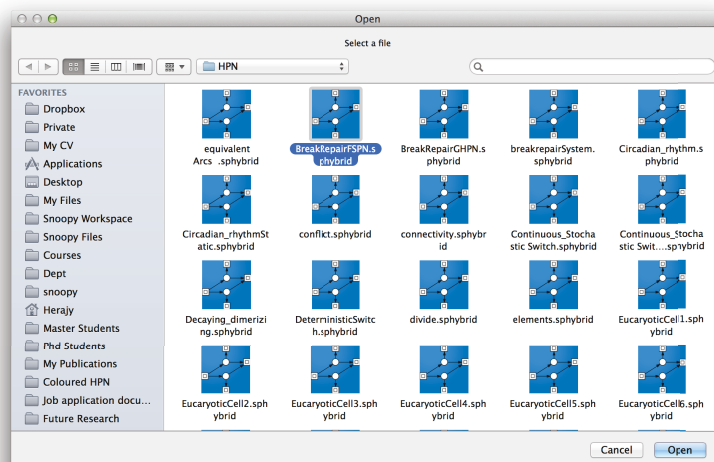


Figure 11: Open dialog.

To open an existing file, follow these steps:

1. From *file* menu, select *open*.
2. The *open* dialog box will appear as in Figure 11.
3. Locate the file name and then click *Open*.

You can also import files written by other formats including *SBML* models, see Section 7.5.

## 5.2 Hybrid Petri Nets

### 5.2.1 Drawing a Net

Let us start by presenting how we can use Snoopy to draw and model the  $\mathcal{HPN}$ . You will learn more about the different elements of  $\mathcal{HPN}$  as well as how to create, edit, and delete them. If you are an experienced Snoopy user, you can skip this section.

### 5.3 $\mathcal{HPN}$ elements in Snoopy

Figure 12 shows the different elements that can be used to model your reaction network using  $\mathcal{HPN}$ . As a matter of organisation we group them here into four cate-

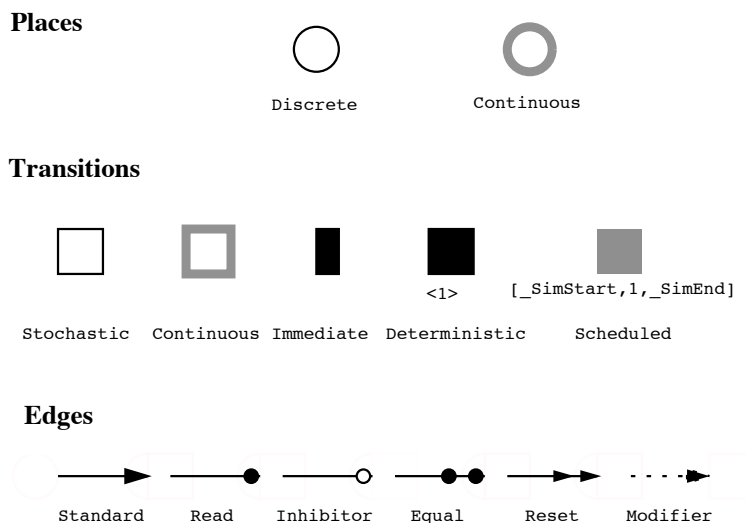


Figure 12: Elements of  $\mathcal{HPN}$  in Snoopy [HH12]. Shown are the default graphical representations, which however can be change in the *Global Preferences*.

gories: places, transitions, arcs, and meta data. Places are used to represent biochemical species, while transitions are used to model reactions which fire and affect the number of molecules/concentration of these species. Arcs connect places to transitions or vice versa, while arcs' weights denote the reaction stoichiometry. Meta data help to organise and annotate your model with text.

In the following, we present a short description of these elements and how they can be drawn in Snoopy.

### 5.3.1 Places

Places are used to model biochemical species. There are two types of places in  $\mathcal{HPN}$ : discrete and continuous. The former type can carry only a non-negative integer number which may represent the species number of molecules. On the contrary, continuous places are used to carry non-negative real numbers that may represent the species concentration. In what follows, we elaborate on these two place types.

1. **Discrete.** If you decide to represent a certain species as a discrete one, then a discrete place (usually drawn as single line circle) can be used to denote such species. This place type will not take part in the corresponding system of ODEs. However,



they can be used to define the transition functions, where a non-standard arc is used to connect this place with continuous transitions. Moreover, the number given inside a discrete place specifies the number of tokens the place hold.

2. **Continuous.** On the contrary, continuous places (usually drawn as a shaded line circle) are used to define the system of ODEs corresponding to the model to be developed. In fact, each continuous place is represented by a variable in the generated ODEs. The numbers inside continuous places are called marks. The place ODE is defined via the rate equations of the continuous pre- and post-transitions of the place.

To draw a place in Snoopy:

- Select the place's type from the element tree by left click on the element type.
- Go to the drawing canvas and left click the mouse button at the position where you want to have the new place.
- The place will be drawn in the Snoopy canvas.

Another way to quickly select an element type is to select the element type from the *element* menu. To those who prefer keyboard shortcuts, Table 1 provides the shortcut keys for all the element types.

### 5.3.2 Transitions

$\mathcal{HPN}$  in Snoopy provide five different transition types: stochastic, continuous, immediate, deterministic, and scheduled transitions. Transitions are used in the biochemical context to represent reactions. For example a reaction in the form  $A + B \rightarrow C$  can be represented by the Petri net shown in Figure 13. The different transition types in  $\mathcal{HPN}$  makes it very easy to model biochemical reactions and simulate them using either deterministic, stochastic or a mix of both approaches. Similar to the process of drawing places, transitions are drawn by selecting the transition type from the element tree followed by a click in the canvas. The different transition types are illustrated in Figure 12. Table 2 provides a comparison between these different transitions. In the following, we briefly summarise the features of each transition type.

1. **Stochastic:** Stochastic transitions which are drawn in Snoopy as a single line rectangle, fire randomly with an exponential distribution delay. The user can specify a

Table 1: Shortcut keys of the different  $\mathcal{HPN}$  elements in Snoopy

Element	Short Cut key
Discrete place	P
Continuous place	Shift + P
Stochastic transition	T
Continuous transition	Shift + T
Immediate transition	Shift + I
Deterministic transition	Shift + D
Scheduled transition	Shift + S
Parameter ( $\mathcal{HPN}^c$ )	K
Coarse place	Ctrl + P
Coarse Transition	Ctrl + T
Standard Edge	E
Read Edge	R
Inhibitor Edge	I
Reset Edge	Z
Equal Edge	Q
Modifier Edge	M
Comment Edge	C
Select	S

set of firing rate functions, which determine the random firing delay. They are intended to represent reactions that require the stochastic semantics. Each stochastic transition is associated with a rate function which is used to calculate the corresponding reaction propensity. The rate can be specified as a free math function or using a predetermined pattern (e.g., mass action). Usually, when defining the rates of stochastic transitions, the transition pre-places are used to specify the mathematical formula. When a stochastic transition fires, it consumes tokens from its pre-places connected with standard arc and adds tokens to its post-places.

2. **Immediate:** Immediate transitions - black bar - fire with zero delay, and they have higher priority in the case of a conflict with other transitions. They may carry weights which specify the relative firing frequency in the case of conflicts between more than one immediate transition. Immediate transitions can be used to implement instant events that fire during the simulation. For instance, when modelling the biological

cell cycle behaviour, immediate transitions can be used to decide when a division should take place as well as implementing dividing each cell component.

3. **Deterministic:** Deterministic (time delay) transitions - black square - fire after a specified time delay. Unlike stochastic transitions, the time delays of deterministic transitions are fixed numbers which is given during model construction. Deterministic transitions can be used to implement events which take place after a pre-specified time delay.
4. **Scheduled:** Scheduled transitions - grey square - fire at a user-specified time point or time interval. Scheduled transitions are special cases of the deterministic ones. You can use scheduled transitions if you need a transition that fires a few times within a specific time period. The user specifies the start and end of the firing period as well as the time period between two successive events.
5. **Continuous:** Continuous transitions - shaded line square - fire continuously in the same way as in continuous Petri nets. Their semantics are governed by ordinary differential equations. Their ODEs define the changes in the transitions' pre- and post-places. Continuous transitions can be used to implement continuous reactions. The rates of continuous transitions are defined in the same way as in stochastic transition, though it has a different meaning during the simulation. In continuous transitions, the rates are used to construct the system of ODEs that corresponds to the continuous part.

### 5.3.3 Arcs

Arcs are used to connect places and transitions. The arc weight is equivalent to the stoichiometry in the biochemical reactions. As a general modelling language, Snoopy *HPN* contains six different arc types: standard, read, inhibitor, equal, reset, and modifier arcs. Figure 12 illustrates the different arc types. Figure 15 shows the possible connections between different place and transition types. Extended arcs (read, inhibitor, equal, reset, and modifier) are used to connect only a pre-place with a transition. The upcoming section provides a more detailed discussion about the connection between places and transitions using those arcs.

1. **Standard:** Standard arcs connect places and transitions in both directions. The arc weight may be a non-negative integer or non-negative real number depending on

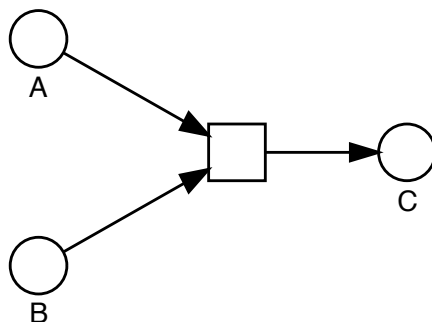


Figure 13: A Petri net representation of a simple reaction  $A + B \rightarrow C$ .

the corresponding place and transition type. A transition connected with this arc type, changes the related place markings when it fires.

2. **Read:** Read arcs do not change the place marking, instead they are used to implement boolean expressions. A transition connected with a read arc can only fire if the marking of the corresponding pre-places are greater than or equal to the arc weight.
3. **Inhibitor:** Similar to read arcs, inhibitor arcs are also used to implement boolean expressions. However, a transition connected with a place via an inhibitor arc can only fire if the corresponding pre-place marking is less than the arc weight.
4. **Equal:** Equal arcs also belong to the category of extended arcs. However, a transition connected with a place using an equal arc can only fire if the arc weight is exactly equal to the pre-place marking. Equal arcs cannot be used to connect continuous transitions, since real values are not easy to compare, unless certain precision is assumed.
5. **Modifier:** Modifier arcs do not affect place marking nor transition enabling. They are used to preserve the net structure when a place is required to be added to a transition rate (only pre-places are permitted in a transition's rate function).
6. **Reset:** Reset arcs are used to reset the connected pre-place to zero when the corresponding transition fires. It does not consume tokens or even enforce a condition on the connected transition.

To draw an arc between a place and a transition, follow these steps:

Table 2: Comparison of the different transition types in Snoopy  $\mathcal{HPN}$ .

Transition	Graphics	Firing	Semantics
Stochastic	Single line rectangle	After an exponential random delay	CTMC
Continuous	Shaded line square	Continuously with time	ODE
Immediate	Black bar	Immediately	
Deterministic	Black square	After a deterministic delay	
Scheduled	Grey square	At a scheduled (absolute) time	Single pulse

1. Select the appropriate arc type from the element tree.
2. Go to the source element (place/transition), then click and move while holding the mouse .
3. Get to the the target element and release the mouse.
4. If the source, target, and the arc fulfil the connection rules (see below), the arc is drawn, else nothing is drawn.

### 5.3.4 Marking-dependent Arc Weights

In addition to using constant values to define arc weights, Snoopy has recently been extended to allow the use of the pre-place to define the corresponding arc weights. For an example, consider the  $\mathcal{HPN}$  in Figure 14 which has been adopted from [HH15]. Here the places X and Y have been used to define arc weights.

### 5.3.5 Meta Data

Meta data refer to other data that are useful to the user, when (s)he draws the model. It does not affect the modelling capabilities but it can enhance the readability and organisation of the modelled Petri nets. As a type of meta data, comments provide a way to add some text annotation to the  $\mathcal{HPN}$  model file. To add a comment to your file:

1. Select "comment" from the element tree.
2. Go to the canvas and click.
3. A comment will be shown with a default text "comment".

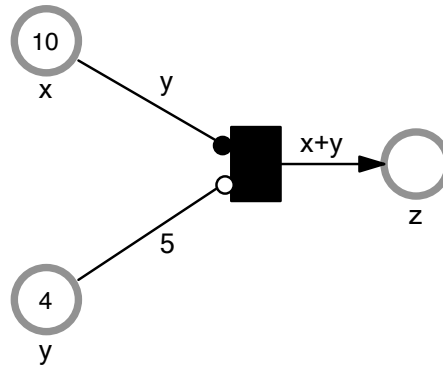


Figure 14: An  $\mathcal{HPN}$  example of using marking-dependent arc weights. The example contains three places  $x$ ,  $y$ ,  $z$  and one immediate transition, which can fire only when the two conditions:  $x \geq y$  (enforced by the read arc) and  $y < 5$  (enforced by the inhibitor arc) hold.

4. Double click the drawn element to change the text.
5. Write your new text and press Ok.

It is worth mentioning here that each element has a comment field. However, this field is element-dependent, which means that if you move this element, the comment moves, too.

### 5.3.6 Connection Rules

The different  $\mathcal{HPN}$  elements are connected with each other, such that they obey certain rules. The connection rules ensure that the underlying semantics of  $\mathcal{HPN}$  elements are still valid. For example, when you try to connect a discrete place with a continuous transition using standard arcs, Snoopy will refuse to carry out such a command. The reason behind Snoopy's refusal is that the semantics of continuous transitions are represented by a set of ordinary differential equations that require the existence of real values in the pre- and post-places. Hence this is not allowed to take place for discrete places. Figure 15 is a graphical illustration of all the connection rules. It is very important to know these rules to use  $\mathcal{HPN}$ ; however you do not have to worry about memorising them since Snoopy will not permit you to violate them.

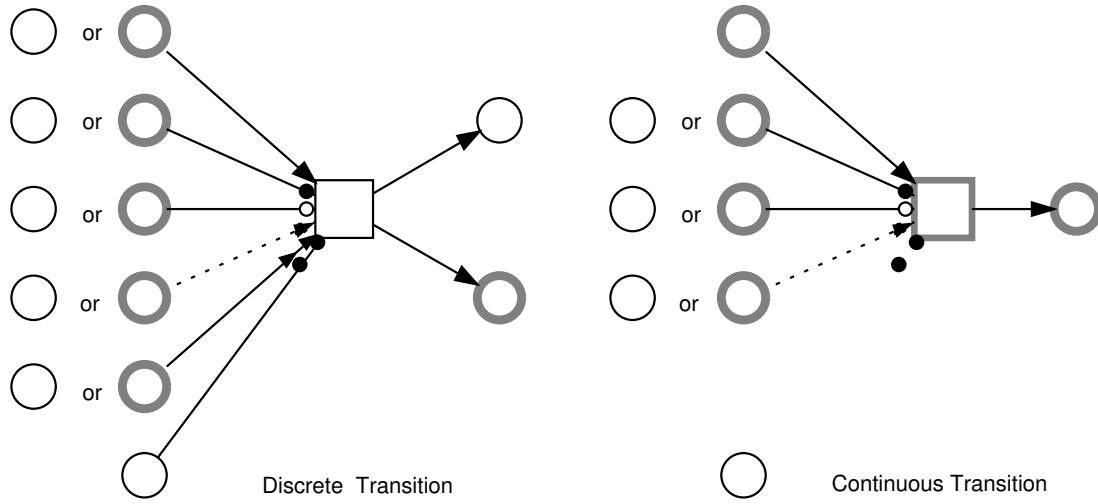


Figure 15: Valid connections between  $\mathcal{HPN}$ 's elements in Snoopy.

Since one of the  $\mathcal{HPN}$  objectives is to bring discrete and continuous parts together, there are some arcs which allow the connections between discrete places and continuous transitions. Read, inhibitor and equal arcs are some examples of these arcs.

However, you are not allowed to make a connection between two similar elements (two places or two transitions), since Petri nets are bipartite graphs. Moreover, extended arcs are used to establish connections between transitions and their pre-places.

#### 5.4 Moving, Editing and Deleting Elements

After the elements are drawn, you can move, edit, and delete them. To move one or more elements, follow these steps:

1. Select the elements you want to move.
2. Left click and hold the mouse while moving to the target position.
3. When you reach the target position, release the mouse button there.

**Tip:** you can also move these elements using keyboard keys. To do so, hit down the shift key and press the *up*, *down*, *left*, or *right* arrow to move the selected elements up, down, left, or right, respectively.

## 5.5 Adjusting Element Properties

Each element (Place, transition, or arc) has a set of properties which can be adjusted during model construction, but not simulation. The type as well as the number of these properties are different from one element class to another. For example, while places and transitions have name attributes, arcs do not. Furthermore, the graphical appearance of each element can be adjusted using these properties. In the following, we provide a brief presentation of these properties.

### 5.5.1 Name

Places and transitions have a name attribute. This attribute has to be unique. If you do not provide a name, a default name is used. For example, the default name for a place is composed as *\_place\_PLACE\_ID*, where *PLACE\_ID* is a unique identifier given to each place automatically. For example, a place with ID =1 has a default name *\_place\_1*. The same rules are applied to transitions; however the word *place* is replaced by *transition*.

To change an element's name:

1. Double click the element of which you want to change its name.
2. A dialog will open like in Figure 16. Enter a new name and click Ok.

Some rules that you should follow while specifying a name for a place or a transition include:

- The name should start with a character and never with a number.
- White space is not allowed.
- Special characters like *?*, *\**, or *#* (except *\_*) are not allowed.
- The name should be unique.

### 5.5.2 Marking

All places have a marking attribute. The default value is zero but you can change this value by editing the marking properties. To edit the marking properties:



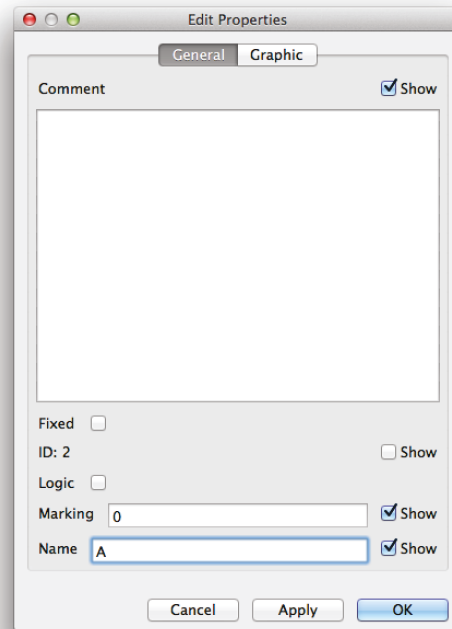


Figure 16: Changing a place's name.

1. Double click the place of which you want to change its marking.
2. The properties box will be shown.
3. Click on the marking tab as shown in Figure 17.
4. Enter the new marking and press Ok.

Please note that discrete places accept only non-negative integer values as marking, but continuous places accept non-negative real values as marking. As an alternative, you can use a constant to specify a place marking. Constants in Snoopy are discussed in Section 5.6.

### 5.5.3 Show/Hide Attribute

Each of the discussed attributes can be shown in the net layout. You can switch on/off showing these attributes by checking/unchecking the corresponding checkbox entitled *show*.

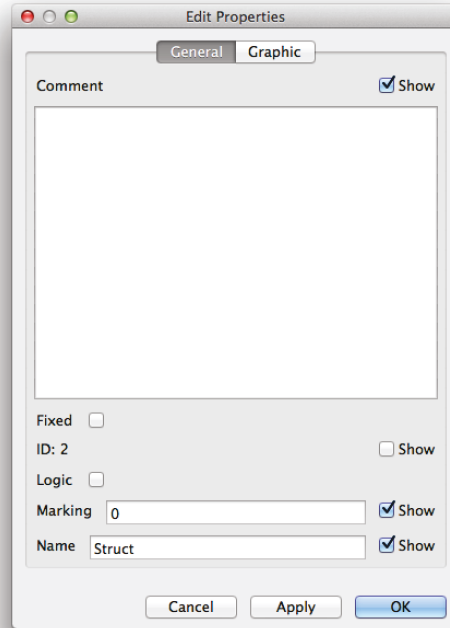


Figure 17: Changing the place's marking.

For example, you can hide the place marking by unchecking the box *show* in front of the marking text box.

#### 5.5.4 Continuous Rates

Each continuous transition has a rate associated with it. This rate function is used to generate the places' ODEs. The transition's rate function is added to the transition's post-places' ODE while it is subtracted from the pre-places' ODEs. The general mathematical formula to generate the ODEs from continuous Petri nets is given by (1)

$$\frac{dp}{dt} = \sum_{t \in \bullet p} f(t, p)v(t) - \sum_{t \in p \bullet} f(p, t)v(t) \quad (1)$$

where  $v(t)$  is the rate function,  $f(t, p)$  is the weight connecting the transition  $t$  with place  $p$ , and  $\bullet p, p \bullet$  are the pre- and post-transitions of place  $p$ , respectively. Note that place names are read as real variables.

Pre-places which are connected with a transition using read, inhibitor, or equal arcs are represented in the ODE as on/off switch.

It is worth mentioning here that the marking on places which are connected with transitions using extended arcs are not affected by the firing of the transitions.

To change the transition rate function, use these steps:

1. Double click the continuous transition of which you want to change its rate function.
2. Select the function tab. Now you can edit the rate function as shown in Figure 18.
3. To get help while editing the rate, click on the function assistant as shown in Figure 19.
4. To check if the formula is correct or not, hit *check* functions.
5. To list the rate function of all continuous transition, click *functions overview*.

In the function assistant, you will be provided with all the transition's pre-places and the constants which have been defined in this net. Moreover, you can select one of the patterns which are defined by Snoopy, for example: *MassAction*, *MichaelisMenten* and so on.

### 5.5.5 Stochastic Rates

While the continuous rates are used to generate the places' ODEs, stochastic rates are used as stochastic delay for the stochastic transitions. The value you enter here will be used as the parameter for the exponential distribution according to which the random numbers are generated.

To edit the stochastic rate, follow these steps:

1. Double click the stochastic transition of which you want to change its rate function.
2. Select the function tab. Now you can edit the rate function as shown in Figure 18.
3. To get help while editing the rate, click on the function assistant (Figure 19).
4. To check if the formula is correct or not, hit *check* functions.
5. To list the rate functions of all stochastic transitions, click *functions overview*.

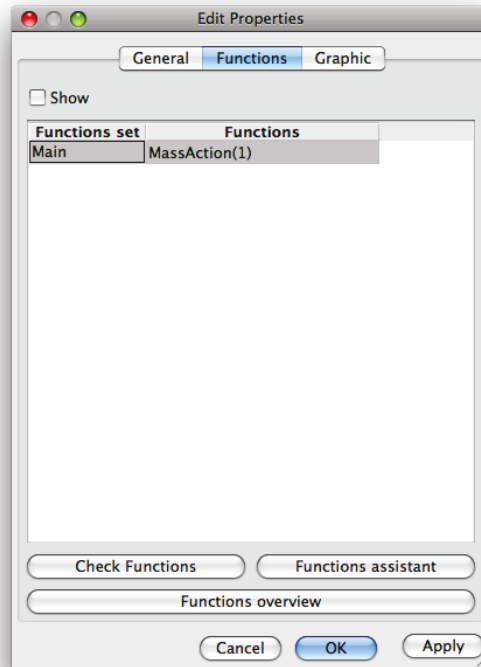


Figure 18: Editing Transition Function.

You will notice that the procedure is the same for editing both continuous and stochastic rates. In fact, Snoopy uses a unified dialog set for editing such common properties in all net classes. Later on you will see that the continuous and stochastic rates can be used interchangeably, when we consider dynamic partitioning of the transitions.

### 5.5.6 Delay

Deterministic transitions fire after a pre-determined time delay. This time has to be specified by the modeller. When a transition is getting enabled, it has to wait for this amount of time before it can fire.

To change the delay of a deterministic transition:

1. Double click the deterministic transition for which you want to edit its delay.
2. Select the delay tab.

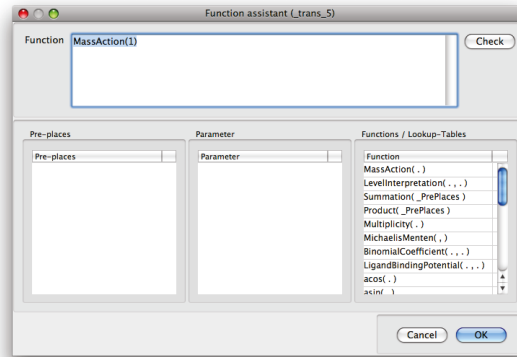


Figure 19: Function Assistant.

3. Enter the new delay value and press Ok.

Also the same principle is applied here to set the delay. Note that a deterministically delayed timed transition does not change its type during the dynamic simulation.

### 5.5.7 Weights

Immediate transitions fire immediately when they are getting enabled. They have priority over all the other transition types, which means that when there is a conflict between immediate transitions and any other transitions type, the immediate transitions will fire first. A special situation occurs, when one or more immediate transitions are in conflict with each other. In this case, one of these transitions is selected based on their weights.

To change the weight of an immediate transition:

1. Double click the immediate transition.
2. Select the weight tab.
3. Enter the new value and press Ok (Figure 20).

Figure 21 provides an example of how the weights of immediate transitions can be used to select one of them to fire. The Petri net contains two immediate transitions,  $T1$  and  $T2$ , and one deterministic transition  $T3$ . The three transitions are in conflict with each other as all of them are enabled. The question is which of them should be selected to fire?

The first rule is that immediate transitions have priority over deterministic ones. This will leave the race between  $T1$  and  $T2$ . Next, we use the weights of  $T1$  and  $T2$  to help in the selection process. The weight of  $T1$  is 2, while the weight of  $T2$  is 1. Thus,  $T1$  will be selected with a probability of  $2/3$ , while  $T2$  will be selected with a probability of  $1/3$ .

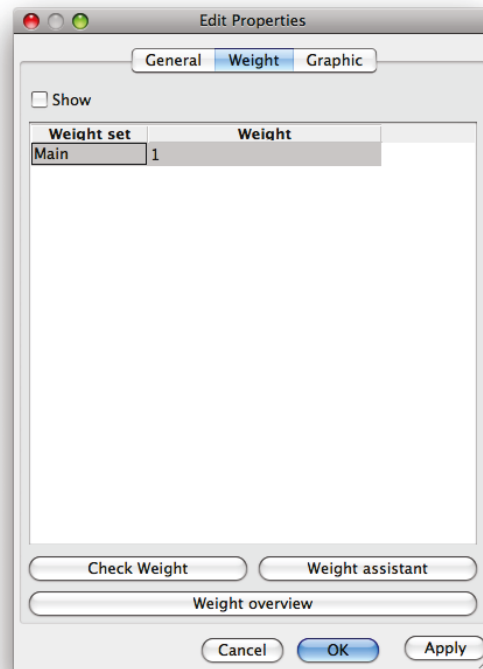


Figure 20: Editing Immediate Transition Weight.

### 5.5.8 Graphics

Figure 12 illustrates the default graphical representation of the  $HPN$ 's elements. As part of Snoopy, All these elements' appearances can be changed. You can select a different line colour, fill colour, or even change the overall element shape as in continuous transitions.

To change the element's graphics:

1. Double click the element for which you want to change its graphics.
2. Select the graphics tab.

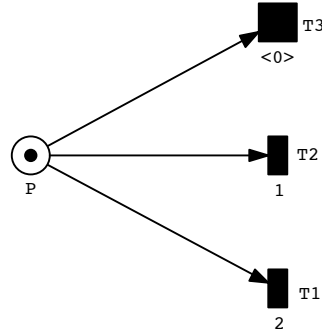


Figure 21: An example of immediate transition weights.

3. To change the line colour of this element, select Pen colour.
4. To change the fill colour of this element, select brush colour.

### 5.5.9 Arc Style

An interesting option related to drawing arcs in Snoopy that might not be obvious for Snoopy newcomers, is the arc style. This will affect the overall appearance of your model. For an example, consider the simple Petri nets in Figure 22. Figure 22a uses the *line* arc style, while Figure 22b adopts the *spline* arc style. Of the two, Figure 22b might look better for your model. When you use Snoopy for the first time, the default arc style is *line*. To change it, follow these steps:

1. From the *File* menu, select *Preference*.
2. The global setting dialog will appear as in Figure 23.
3. Click on the *elements* tap and navigate to *edge style*.
4. Select between *line* or *spline* style and hit *ok*.
5. Your selected edge style will be changed for all arcs in your whole net.

Please note that in the *global setting* dialog, there are many other options that you can change in order to customise Snoopy to fit your needs. A detailed discussion about each feature is beyond the scope of this manual. However, you can easily try each of them. In addition, the location of the *Preference* depends on your operating system.

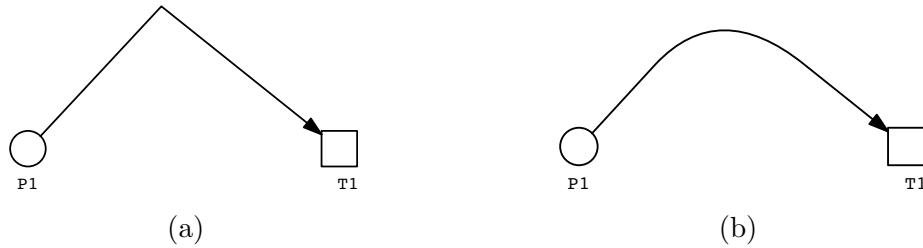


Figure 22: Comparing the different edge styles in Snoopy: (a) line style and (b) spline style.

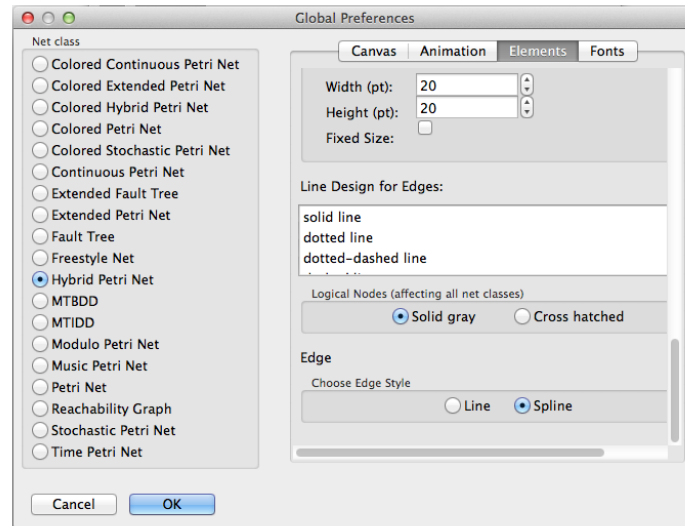


Figure 23: Changing the setting of the edge style.

### 5.5.10 Other Properties

There are many other useful features and properties which  $HPN$  inherit from Snoopy. Some of these are: searching for a node, sorting nodes, squeezing node numbers, checking for duplicate names, and a long list of such things which we can not cover here. If you are interested, please consult some introductory materials about Snoopy.

### 5.5.11 Converting Between Elements

To make the process of constructing  $HPN$  user-friendly, Snoopy permits the change of elements from one type to another type from the same groups. That is discrete places can be converted to continuous ones and vice versa. However, places cannot be converted into



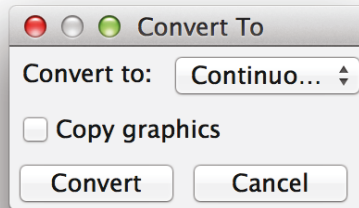


Figure 24: The *Convert to* dialog.

transitions. This option is very useful, if you change your mind after the element has been fully drawn and connected with other elements. To convert an element from one type to another, follow these steps:

1. Select the element (s) you want to convert. They should belong to the same element class.
2. From *edit*, select *Convert to*.
3. The *Convert to* dialog will appear as shown in Figure 24 with the possible options for conversions.
4. Select the new element type and click *Convert*.
5. The selected items will be converted to the specified type.

You can also check the option *Copy Graphics* to preserve the same graphics attributes. Please note that if the conversion process will violate the connection rules discussed in Section 5.3.6, then Snoopy will not permit such conversion. In this case you will get a message in the *log* window. Furthermore, this option can also be applied to convert from one arc type to the others, with the exception of *Modifier arcs*.

## 5.6 Constants and Functions

The constant section in Snoopy is used to define constants and assign them values such that they can be (re)used when defining transition rates, place markings, or arc weights.

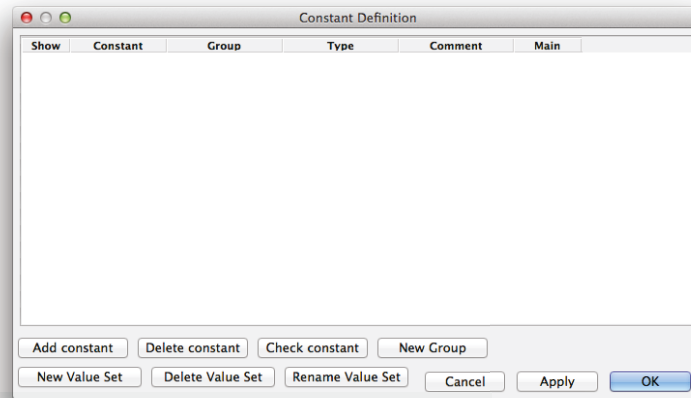


Figure 25: The constant dialog box in Snoopy.

The function sections is used to define simple functions so that they can be used later to define transition rates and the initial marking. At the time of writing this manual, Snoopy does not support the use of functions in hybrid Petri nets, but it is planed to implement such feature.

To open the constant dialog, please follow these steps:

1. On the Snoopy main menu, click on *constants*.
2. The constant dialog box will open as shown in Figure 25.

### 5.6.1 Defining a New Constant

To define a new constant:

1. On the constant dialog, click on the *add* button.
2. A new row will be added.
3. Enter a valid name, select the group name and constant type, and give it a value.
4. A constant will be added as shown in Figure 26.

Please note that:

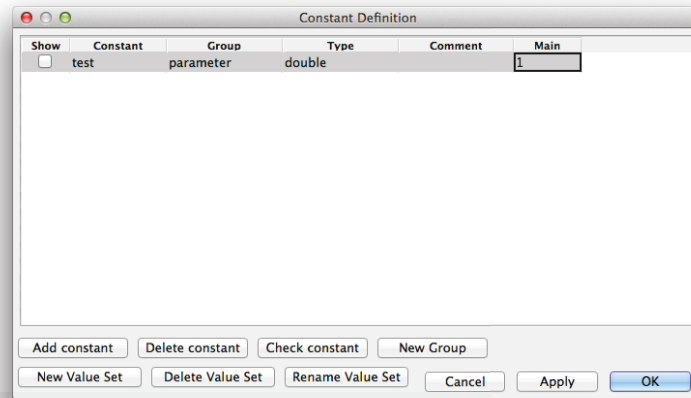


Figure 26: Adding a new constant in Snoopy.

- Constant names should obey the same rules as places and transitions (see Section 5.5.1).
- Groups are used to combine a set of constants so that we can deal with them as one entity during the simulation.
- The constant type specifies the type of the constant (currently, *int* or *double*).
- You can add comments in the *comment* field.

### 5.6.2 Defining a New Constant Group

By default, just two groups of constants are available: *marking* and *parameter*. However, you can easily add more constant groups. To define a new group, please follow these steps:

1. In the constant dialog, select *New Group*.
2. A small dialog will open asking for the group name.
3. Enter a name and then click *ok*.

After adding a new group, you can use it to assign constants you to one of them.

### 5.6.3 Deleting a Constant

Sometimes after you added a constant, you feel that you do not need it any more. To delete a constant, just select the corresponding constant row and then click on the *delete constant* button.

### 5.6.4 Manipulating Constant Value Sets

In Snoopy you can assign more than one value to the same constant but not concurrently. Later during the simulation, you can select which value set to use for such group of constants. The value set appears as a new column in the constant dialog. The default value set is *main*, you are not allowed to delete or rename it. You can add as many value sets as you want. To add a new value set, follow these steps:

1. On the constant dialog, click the *New Value Set* button.
2. A new small dialog will open.
3. Enter a name and click *Ok*.
4. The new value set will be added. Give values to the constants you would like to use in combination with this value set.

To delete a value set:

1. Select the column representing the value set you would like to delete.
2. Click on *Delete Value Set*.

### 5.6.5 Using Constants in Element Properties

Once constants are defined, you can easily use them to define the element properties, including initial marking, arc weights and transition rates. This will be very useful to replace numerical values.

## 5.7 Coloured Hybrid Petri Nets

Coloured Hybrid Petri Nets ( $\mathcal{HPN}^c$ ) are a coloured version of  $\mathcal{HPN}$ .  $\mathcal{HPN}^c$  extend the power of  $\mathcal{HPN}$  and  $\mathcal{PN}^c$  by combining the power of the two Petri net classes into

one class. Thus they can flexibly be used to address the two main challenges of multi-scale modelling which require the interplay between discrete and continuous semantics. The formal definition as well as the formal semantics of  $\mathcal{HPN}^C$  can be found in [HLR14].  $\mathcal{HPN}^C$  can be automatically unfolded to  $\mathcal{HPN}$ , so the simulation of  $\mathcal{HPN}^C$  is done on an automatically unfolded  $\mathcal{HPN}$ .

To demonstrate the use of  $\mathcal{HPN}^C$ , we consider a simple example, the repressilator [LH14]. The  $\mathcal{HPN}^C$  model is illustrated in Fig. 27, and its unfolded  $\mathcal{HPN}$  model is given in Fig. 28. The partition of continuous and discrete nodes is given as follows. The 1-bounded places as determined by P-invariant analysis and the related transitions as determined by T-invariant analysis are kept discrete. The unbounded places and related transitions are approximated by continuous places and transitions, respectively. That is, places *gene<sub>i</sub>* and *blocked<sub>i</sub>*, and transitions *block<sub>i</sub>* and *unlock<sub>i</sub>* are treated as discrete, and all other nodes as continuous, where  $i = a, b, c$ .

In the following, we will illustrate how to construct a  $\mathcal{HPN}^C$  model using Snoopy. More detailed instructions how to construct a general colour Petri net can be found in [LHR12].

### 5.7.1 Defining Colour Sets

To build an  $\mathcal{HPN}^C$  model, we first need to define colour declarations, which consist of colour sets, constants, variables, and functions. In Snoopy, we offer specific tables in dialogs for accomplishing the declarations. Alternatively, the colour declarations required to construct a coloured Petri net can be provided by a *CANDL* file. A *CANDL* file can be edited with any text editor, and might be more convenient for the experienced users. For more information about the specification of *CANDL* language and file format, please consult [LHR12].

As part of specifying the coloured model, one need to define colour sets. Each colour set represents a group of similar objects, e.g., cells, or tissues. In Snoopy, we offer both simple types, *Dot*, *int*, *string*, *enum* and *index*, and *compound types*, *product* and *union*, for defining colour sets. Compound colour sets are based on simple colour sets that are already defined.

When defining colour sets in Snoopy, we can use the following steps:

- In the declaration panel (see Figure 29), double click *Simple color sets*, and then a colour set definition dialog appears (see Figure 30).

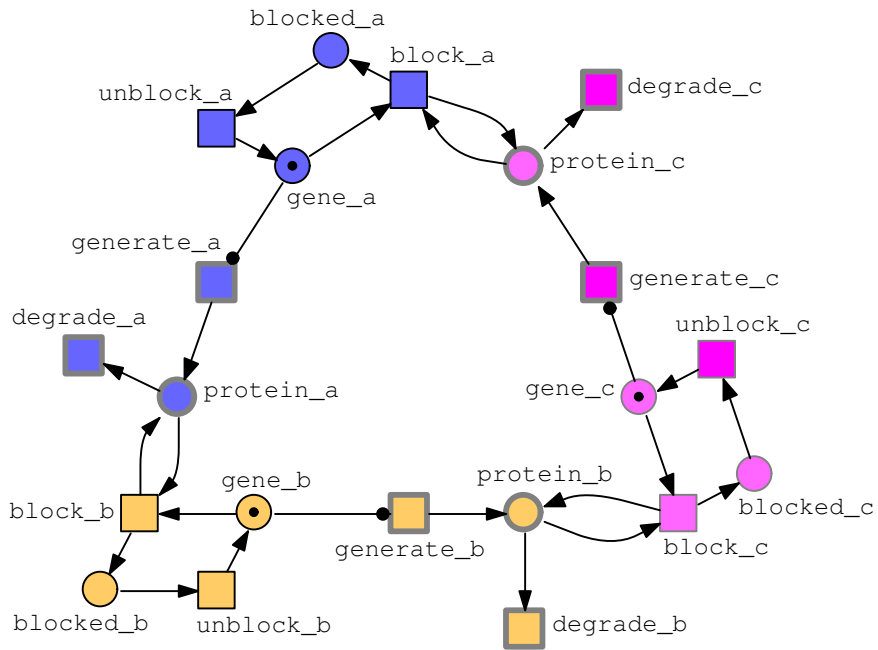


Figure 27: An  $HPN$  model of the repressilator. In this model, there are three genes forming a circle through inhibition, i.e.,  $gene_a$  inhibits  $gene_b$ ,  $gene_b$  inhibits  $gene_c$ , and  $gene_c$  inhibits  $gene_a$  [LH14].

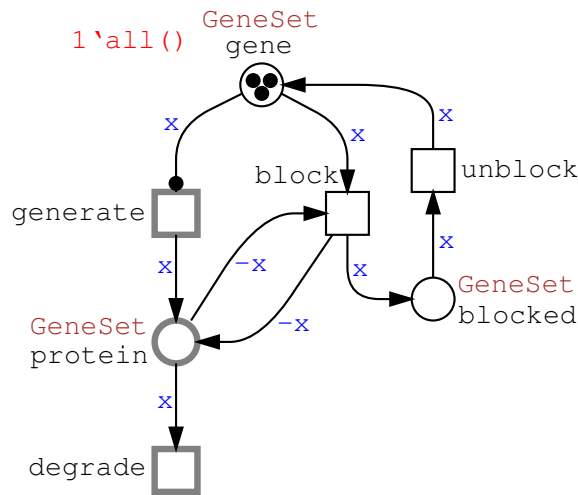


Figure 28: An  $HPN^c$  model of repressilator, obtained by folding the three genes into a colour set of three colours. Declarations: colourset  $GeneSet = \text{enumeration with } a, b, c;$  variable  $x: GeneSet$  [LH14].

- In the dialog, you can add or delete a colour set. If you add a new colour set, you have to provide a name, choose a type and define the colours you want. For example, here we define a colour set GeneSet of the enum type with three colours, a-c.
- However, if you want to define compound colour sets, you can click *Compound color sets* and then a similar dialog appears (see Figure 31). There, you also need to provide the name, choose a type (product or union) and choose the defined colour sets, based on which you define your compound colour set. If necessary, you can also use predicates to choose a subset of a colour set.

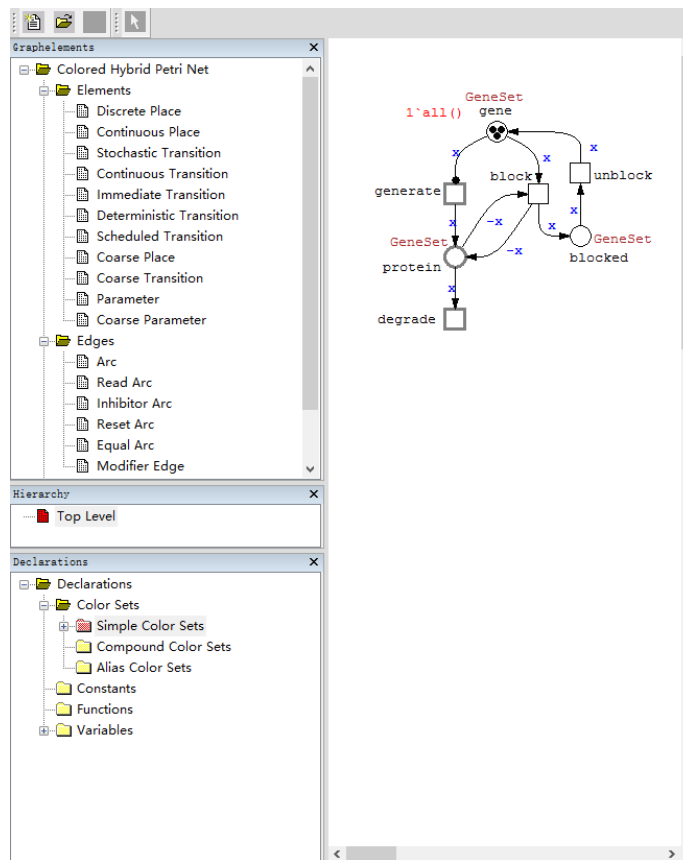


Figure 29: The declaration panel.

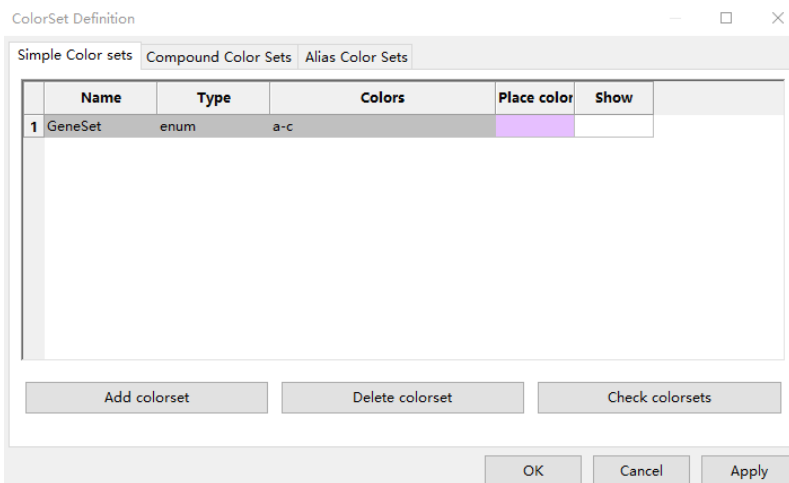


Figure 30: Simple colour set definition dialog.

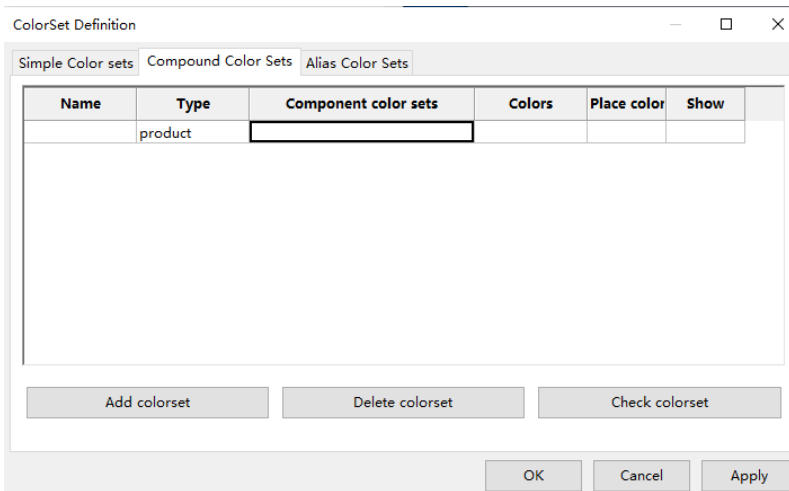


Figure 31: Compound colour set definition dialog.



### 5.7.2 Defining Constants

Constants can be used throughout the model, which facilitate the scalability of the model by only modifying the values of constants. The steps to define a constant are as follows:

- In the declaration panel, double click the "Constants" tab, and then a constant definition dialog appears (Figure 32).
- Here, you have to provide the name, choose a type, e.g., the most widely used type is int, and then set the value for a constant. For example, in Figure 32, a constant  $N$  is defined as the integer type and has the value 3.
- Besides, you can add or delete a constant or check the syntax of already defined constants.

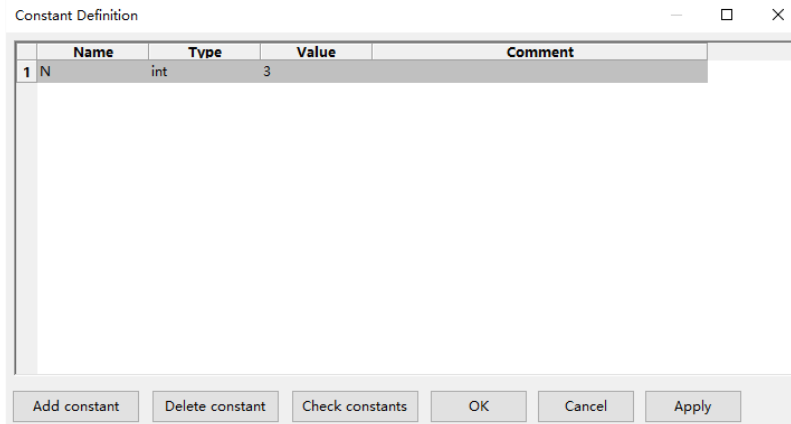


Figure 32: Constant definition dialog.

### 5.7.3 Defining Variables

We can also define variables, which are always associated with specific colour sets. Variables are widely used throughout a model, e.g., each arc may have an expression with a variable. The steps to define a variable are as follows:

- In the declaration panel, double click the *Variable* tab, and then a variable definition dialog appears (see Figure 33).

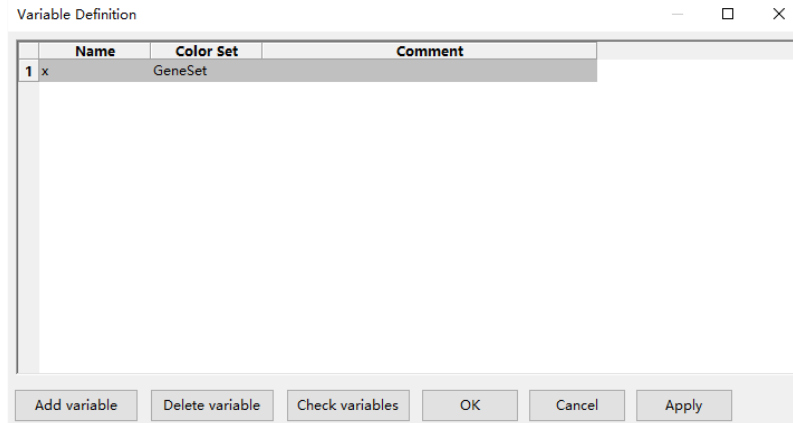


Figure 33: Variable definition dialog.

- Here, you have to provide the name of a variable and choose the colour set of the variable. For example, here a variable  $x$  is defined with the colour set *GeneSet*.
- Besides, you can add or delete a variable or check the syntax of the already defined variables.

#### 5.7.4 Annotating Elements with Colours

When all declarations are given, the next step is to assign them to the net to be constructed. Take the net in Figure 29 as an example. Here are the main steps:

- (1) Assign colour sets to places and define the initial marking of the net.
  - Double click a place, and a property dialog appears.
  - Click the Markings tab (see Figure 34). We can choose a defined colour set for the place in the colorset region.
  - Define initial tokens for this place. In the MarkingList region, the first column gives the colours and the second one gives the multiplicity of these selected colours. For example, here `all()` means 'all colours in the colour set *GeneSet*'. That is, here a token for each colour is assigned to the place as the initial tokens.
  - Besides, you can also check the syntax of the marking definition or obtain an overview of the initial tokens for all places by clicking the *Marking overview* button.

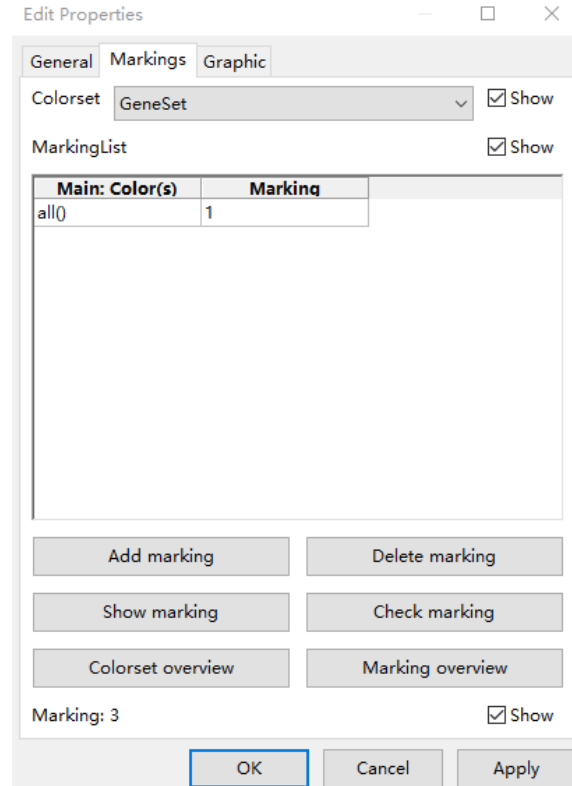


Figure 34: Place property dialog.

(2) Assign guards to transitions.

- Double click a transition, and a property dialog appears.
- Click the *Guards tab* (see Figure 35). Now, we can define a guard for the transition. For example, here the default guard *true* is assigned to the transition, but is not given explicitly.
- If we click *Guard assistant*, we can use the assistant to define a guard.
- We can also check all legal transition instances by clicking the *Show binding* button.

(3) Assign expressions to arcs.

- Double click an arc, and a property dialog appears.



Figure 35: Transition property dialog.

- Click the Expression tab (see Figure 36). We can define an expression for the arc. For example, here the expression for this arc is given as  $x$ .
- If we click the *Expression assistant* button, we can use the assistant to define an expression.
- Besides, we can also observe the expressions for all the arcs by clicking the "Expression overview" button.

If all the steps above are done, we basically obtain a complete colour net. If the net passes the syntax check (*Edit-Check nets...* in the main menu), we can run simulation for this net. More details about the construction of coloured Petri nets can be found in the coloured Petri net manual [LHR12]. For readers who want to know more about  $\mathcal{HPN}^c$ , we strongly suggest to carefully read this manual and the colour-specific manual in [LHR12].

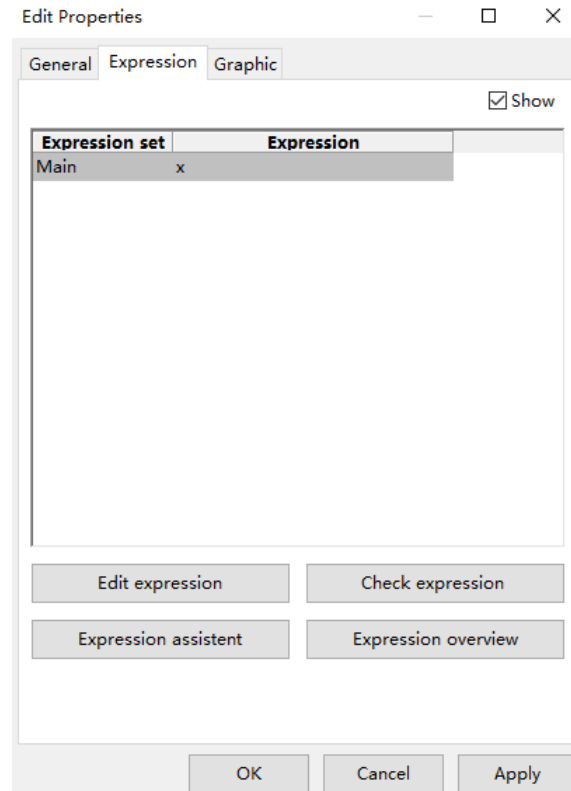


Figure 36: Arc property dialog.

## 6 Model Simulation

Once an  $HPN$  model has been created, it can be executed via the Snoopy simulator. In what follows, we discuss the main steps in simulating a hybrid model created using Snoopy.

### 6.1 Launching the Simulation Dialog

To open the simulation dialog, follow these steps:

1. From the *View* menu, select *Start Simulation-Mode*.
2. If you are using  $HPN^c$ , the unfolding dialog (Figure 37) will appear. In this case, select an unfolding algorithm (or accept the default one), then click *Start*.
3. The simulation dialog will appear as shown in Figure 38.

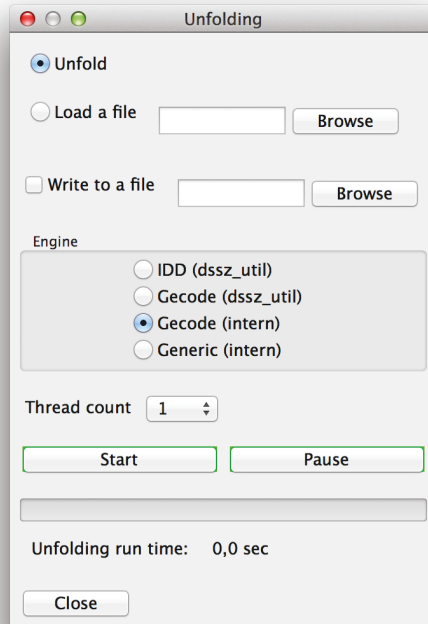


Figure 37: A screenshot of the unfolding dialog.

Please note that the unfolding process may take some time, depending on your model size. If you encounter a problem while opening the simulation dialog, please check the FAQ in Section 9.

The simulation dialog consists of the following sections:

1. **Model configuration:** a set of options that permit to configure the model (e.g., changing the initial marking, or changing constant values).
2. **Simulator Configuration:** a set of options that permit to customise the hybrid simulator.
3. **Export/import details:** help to adjust the import and export options.
4. **Views:** define and edit result views.
5. **Simulation:** start and stop the simulation.

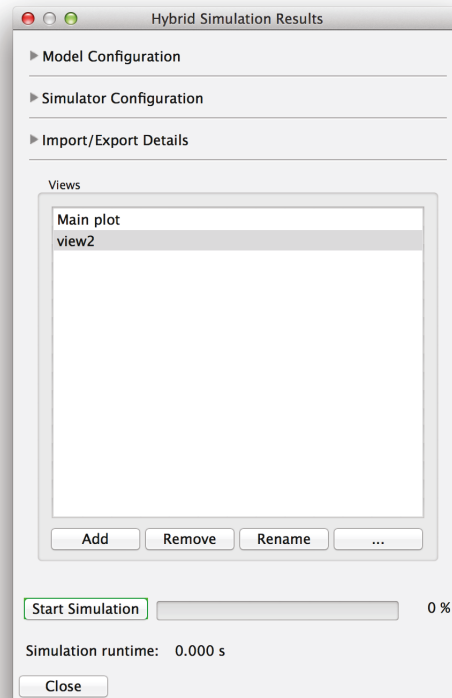


Figure 38: A screenshot of the simulation dialog.

A detailed discussion of each sections is presented in the subsequent sections.

## 6.2 Starting the Simulation

To start the simulation admitting the default setting, just hit the *Start* button. The simulation progress will be shown in the *progress bar* in front of the *start* button. After the simulation finishes, you get the simulation runtime displayed below the *Start* button. If you did not notice any progress in the simulation for a long period, please read the FAQ in Section 9.

### 6.3 Model Configuration

The associated data with the Snoopy model can be configured from the simulation dialog. Examples of model data that can be changed are: initial marking, transition rates, and constant values. To change the model configuration:

1. Click on the *model configuration* collapse control.
2. Adjust the item(s) you would like to change.
3. Your changes will take place next time you run the simulation.

Below is a summary of the different settings that you may need to change. For many models, you can simply accept the default settings.

#### 6.3.1 Adjusting the Initial Marking

To change the current marking:

1. In front of *Marking overview*, click modify.
2. The marking overview dialog will open.
3. Change the required values and hit *ok*.

#### 6.3.2 Adjusting Transition Rates

As  $\mathcal{HPN}$  contain different transition types, you may need to adjust such values from the simulation dialog. In fact you can either select a different rate set or modify existing rate values. To select a rate set:

1. Locate the transition type for which you would like to change its rate.
2. Select a rate from the list.
3. Your changes will take effect next time you run the hybrid simulation.

Please note that you have first to define different rate sets, while you are designing the net, in order to select from them.

To modify the rate of a specific transition group:

1. Locate the transition of which you would like to change their rate.



2. Click *modify* in the front of the transition group rate.
3. Change the rate and then click *ok*.

The tool presented in this section helps you to try different rates while simulating your model, without going back to the model editing and adjusting the rate values. For instance, if you would like to check the model behaviour under the rate constant values of 0.1, 0.01, and 0.001, you can simply define rate sets corresponding to these values. During the simulation, you will only need to choose the corresponding rate set to make a test.

### 6.3.3 Adjusting Parameters

Similarly, the different constant values you define in the *Constant definition* dialog in Section 5.6 can be manipulated during the simulation. You can either select from different constant groups or you can directly access the *Constant definition* dialog. To select a different constant group, follow these steps:

- In front of *Parameter*, select a different set.
- Your changes will take effect next time you run the simulation.

In addition to the default constant group *parameter*, you can define other constant groups in the *constant definition* dialog so that you can select among them during the simulation. For example, you can define a group called *arc weights*, that define arc weight constants.

To access the *Constant definition* dialog so that you can directly change constant values:

- In front of the *Parameter* constant set, select *Modify*.
- The *Constant definition* dialog will appear.
- Adjust the required values and hit *Ok*.

## 6.4 Simulation Configuration

Instead of using the default simulation settings, you can adjust them to fit your specific model requirements. To show the simulation settings, click the collapse control called *Simulation configuration*. The simulation options will appear as depicted in Figure 40.

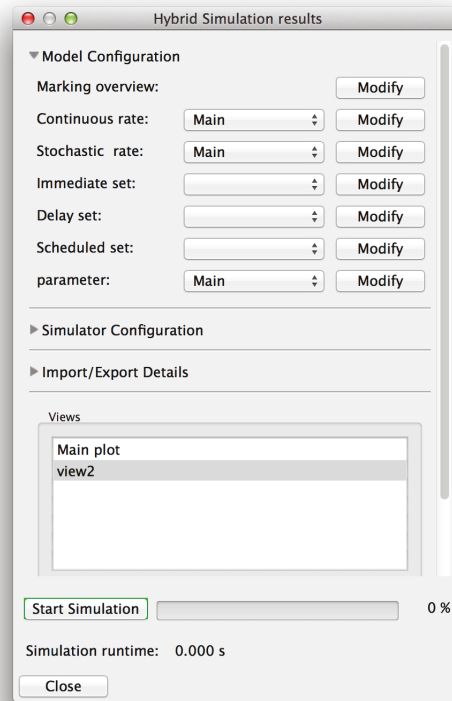


Figure 39: A screenshot of configuring a model from the simulation dialog.

All the options related to the simulation configurations can be grouped into three parts: simulation intervals, ODE solver settings, and time synchronisation settings. In what follow, we discuss them in more details.

#### 6.4.1 Simulation Intervals

This includes:

- **Interval Start:** define the time point from where the simulator begins to record the output. In all cases the simulator will start the simulation at time zero. But the actual point the output will be recorded is *Interval Start*.
- **Interval End:** define the time point where the simulation stops.

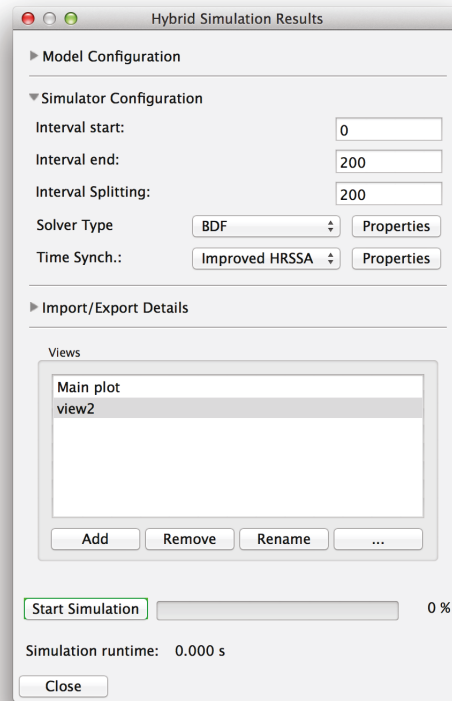


Figure 40: Simulation options.

- **Interval Splitting:** define the total number of output points. This option also specifies the output resolution. Please note that if you specify  $N$  points, you will get  $N + 1$  points. The extra point will represent the initial system state (initial marking).

To alter any of these options, simply change the corresponding text box value before starting the simulation. The default values for *Interval Start*, *Interval End*, and *Interval Splitting* are 0, 100, 100, respectively. Furthermore, these values are saved in the Snoopy file so that when you close and open the Snoopy file, you will get such values loaded.

#### 6.4.2 ODE Solver Settings

Snoopy's hybrid simulator uses internally SUNDIAL CVODE [HBG<sup>+</sup>05] to solve the system of ODEs due to the deterministic part of the model. There are a few options that you may need to adjust in order to improve the performance of the overall hybrid simulator. All

of the options listed below are highly model-dependent and there are no optimal settings that can work for all kind of models.

Snoopy supports three main algorithms for ODE numerical integrations. You can select among them, if the default ODE solver does not fit your model requirements. In the following is a list of these solvers.

- *ARK* (Adaptive Runge–Kutta) can be used to simulate stiff/nonstiff models using the Runge–Kutta algorithm. For models which do not contain many places ( $< 1000$ ) and are not very stiff, ARK can offer a good performance.
- *BDF* (Backward Differentiation Formula): this solver can deal with very stiff models. It is selected by default. For the majority of models, BDF is the best choices.
- *Adams*: this is a solver for non-stiff models. You can use it for simple models.

To change the current ODE solver:

1. Locate the *Solver Type* option at the simulation dialog.
2. Select the one you desire from the list.
3. Your change will take effect the next time you run the hybrid simulation.

In addition to the type of ODE solver, you can adjust the options associated with the selected solver. The different options are shown in Figure 41. In what follow, we summarise the purpose of each field.

- **Initial step size**: the initial step that the ODE solver begins with. This value should be below 1 and above 0. The default value is 0.1. **However, for stiff models, you should use smaller values (e.g., 1e-10 or even smaller).**
- **Type of linear solver**: The ODE solver uses a linear solver to solve the system of linear equations due to the numerical integration algorithm. Therefore, you should select a linear solver from the list. Available options in Snoopy are:
  1. CVDense
  2. CVDiag
  3. CVSpbcg

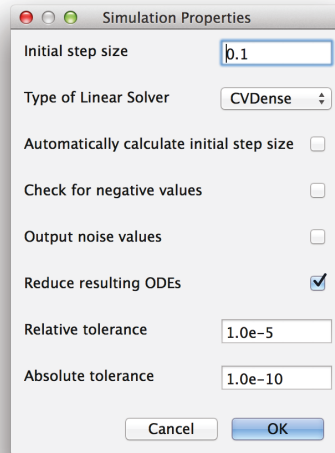


Figure 41: The different settings of the ODE solver.

#### 4. CVsptfqmr

Generally speaking, you should not be concerned with the type of the linear solver. The default one, *CVDense*, usually can fulfil your needs. However, for bigger models (> 2000 places), you may use *CVSpgb*. For very simple models (10 to 200 places), *CVDiag* can result in a better performance. For more information about these linear solvers, you can contact the SUNDIAL user manual [CVO].

- **Automatically calculate initial step size:** if you do not want to provide the simulator with an initial step to start the integration with, then you can check this option so that the ODE solver will employ a special algorithm to select the best initial step. However, based on our experience, the ODE solver takes a considerable time to automatically select the initial step. Therefore, depending on your model, it would be better to try the best initial step and provide it explicitly to the ODE solver. As a recommendation, turn this option *on* for small models, while specifying the initial step size explicitly for bigger models.
- **Check for negative values:** in certain cases, the model might produce negative values, maybe due to incorrect model structure or inappropriate rates. Turning this option

*on*, you can get a warning if such behaviour occurs. This option is useful during the experimental phase.

- **Output noise values:** for certain models, the ODE solver might produce tiny negative values. These values can be neglected and do not influence the model correctness. To prevent such values, uncheck this option.
- **Reduce resulting ODEs:** the system of ODEs might exhibit redundancy due to the model structure (e.g., reversible reactions). Turning this option *on*, will tell Snoopy to reduce the resulting system of ODEs and use a compact internal representation. If this option is turned on, the reduced system of ODEs will be exported to a text file.
- **Relative tolerance:** is used to control relative errors. The default value is 1.0e-5.
- **Absolute tolerance:** is used to control absolute errors. The default value is 1.0e-10.

Both the relative and absolute tolerance are problem-dependent and should be appropriately chosen to get an accurate solution. One idea is to do a few experiments with the tolerances to see how the computed solution values vary as tolerances are reduced. For more information about these two measures please consult the SUNDIAL user manual [CVO].

To change one of these options, select *Properties* in front of the *Solver Type*. The properties dialog box shown in Figure 41 will appear. Change one or more options and click *Ok*. Please note that you cannot alter these properties while the simulation is progressing.

### 6.4.3 Time Synchronisation Setting

There are six simulation algorithms that you can select from to execute your  $\mathcal{HPN}$  model. These include:

- **Static (exact):** uses the original idea of Haseltine and Rawlings from [HR02] to perform the simulation. Although this method is very accurate, it is very slow. Please use it only for simple model.
- **Static (accelerated):** An accelerated version of the hybrid simulation algorithm that makes use of loosely coupled connections between stochastic and deterministic parts to improve the simulation performance. For more information, see [HH16].

- **Improved HRSSA:** this method combines the accelerated method with the hybrid rejection-based stochastic algorithm from [MPT16]. Currently, this is the fastest hybrid algorithm supported by the Snoopy simulator.
- **Dynamic:** this algorithm performs the partitioning dynamically while the simulation is progressing.
- **Continuous:** choosing this simulator will enforce the simulator to read the entire model as a deterministic one. In other words, the model is completely simulated deterministically.
- **Stochastic:** choosing this simulator will enforce the simulator to read the entire model as a discrete and stochastic network.

Moreover, there are a few options that you can adjust to customise the adopted hybrid solver. They are shown in Figure 42. Below is a summary of their meaning:

- **Visualise in between results:** turning this option *on* will ask Snoopy to draw the result while the simulation is progressing. You can turn this option *off*, if you are interest only in the final result.
- **Refresh rate:** define the frequency of how often Snoopy will refresh the simulation result (in millisecond). The default value is 5000 ms. You can ignore this option, if you turn *Visualise in between results* off.
- **Number of runs:** define the number of runs that Snoopy performs. The default is one.
- **Number of threads:** If you would like to perform multiple runs simultaneously, you can specify the number of threads (cores) to use. The default is one (meaning sequential). The maximum number of threads, that you can use, is the number of cores available on your computer.

## 6.5 Viewing Simulation Results

After the simulation finished, you can explore the model results. Snoopy supports *views* to make it easy for you to explore the result from different perspectives. A *view* is a collection of places or transitions of which you would like to explore their behaviours. In addition, a *view* is associated with a *viewer* that is used to visualise the numerical values. The viewer can be a simple xy plot or just a tabular list of the curve values.

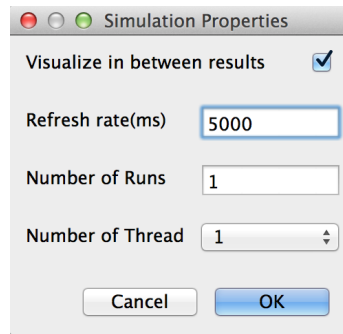


Figure 42: Options for customising the hybrid simulator.

### 6.5.1 Opening the default view

When you open the *simulation dialog* for the first time, Snoopy creates a default view. To open the default view, just *double click* the view name. The default view will be opened as shown in Figure 43.

A view opened for the first time is empty without any places or transitions selected. You can select some of the model places to monitor their values. To do that, follow these steps:

1. From the *View* dialog, click on the *Edit Node List* button.
2. The *Edit Node List* dialog will open as in Figure 44.
3. Use the buttons in the middle to move nodes from the available list to the selected list.
4. To select transitions instead of places, select the required node type from the *Observers*.
5. When you finish, hit *Save*.
6. The curves of the selected items will appear as shown in Figure 45.

Please note that the *Edit Node List* dialog is a little bit different when using  $\mathcal{HPN}^c$ , where a few options are added to enable the selection between folded and unfolded nodes. Moreover, you can use a regular expression to define the set of selected nodes. Regular expression facilitate the selection by just typing a few characters that specify some shared



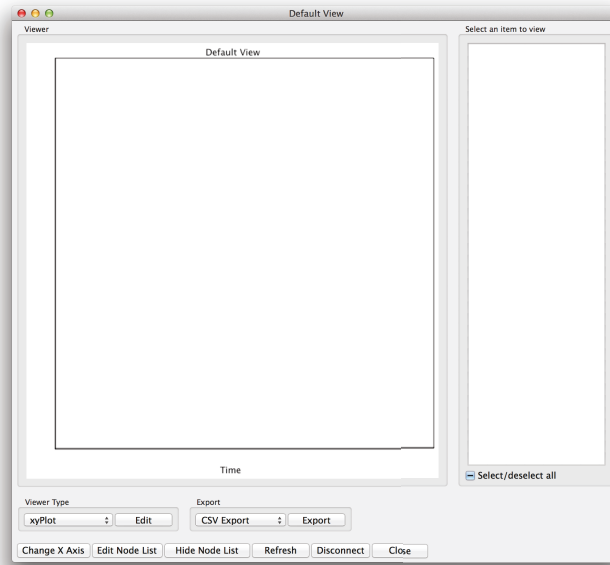


Figure 43: Opening the default view for the first time.

characteristics of the node names to be selected. To use the regular expression to select some nodes, follow these steps:

1. On the *Edit node dialog* locate the text box called *RegEx*.
2. Type the regular expression characterising the nodes that you would like to select.
3. While typing you will notice that some nodes are moved from the available list to the selected one, or vice versa.
4. To select all available nodes, just type ”.”.

For more information about the *RegEx* feature in Snoopy, please consult [Aga15].

### 6.5.2 Manipulating Views

You can adjust the view setting by changing the current viewer as well as changing the curve colours and styles. To change the *viewer* of the current *view*, follow these steps.

1. Under the *viewer type*, select a different viewer from the list.

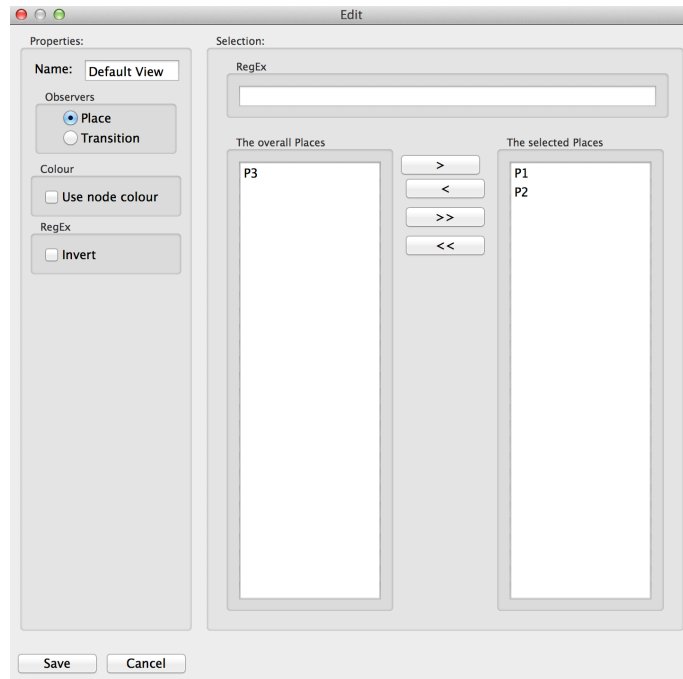


Figure 44: The edit node list dialog.

2. Available options are: *tabular*, *xyPlot*, and *histogram*.
3. The current view will immediately reflect your selections.

Currently, Snoopy supports three types of viewers: tabular (see Figure 46), XY-plot (see Figure 45), and histogram plot (see Figure 47). A tabular viewer shows the data in a table with the first column representing the time and the other columns representing the values of the selected nodes. A XY-plot plots the data such that the values of one variable will be in the X axis and the other selected node values will be on the Y-axis. The X-axis represents by default the time. However it can be adjusted to represent a place or a transition of the Petri net. The final type is the histogram viewer; it shows the simulation results as a histogram where the frequencies of node values are plotted.

### 6.5.3 Exporting Views

Views in Snoopy are intended to give the user a general feedback about the simulation results. However, views can be exported into other formats such that they can be manipulated or carefully visualised using other specialised software. Views can be exported into

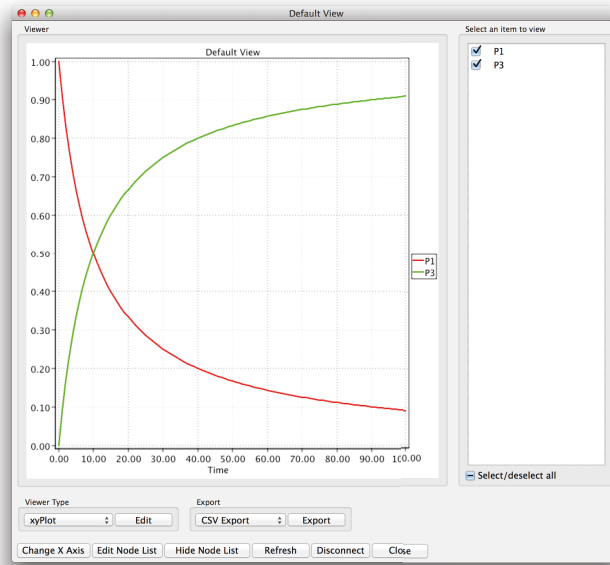


Figure 45: Default view after selecting two places.

either CSV (comma separated values) file format or as image files. To export a *view* into a CSV format, follow these steps:

1. On the *View* window, click *Export* in front of *CSV Export*.
2. The *Export Properties* dialog will open as shown in Figure 48.
3. Enter the file name as well as *the spacer* (see below for details) and hit *Export*.
4. The current view is exported and a message in the log window will be added to inform you about the export result.

Please note that the *spacer* field enable you to specify the separator between two values in the CSV file. Available options are:

- Colon, a ":" symbol will be used.
- Comma, a "," symbol will be used.
- Semicolon, a ";" symbol will be used.

	P1	P2	P3
0	1	0	
1	0.909091	0.0909092	
2	0.833333	0.166667	
3	0.769231	0.230769	
4	0.714287	0.285713	
5	0.666668	0.333332	
6	0.625002	0.374998	
7	0.588237	0.411763	
8	0.555558	0.444442	
9	0.526318	0.473682	
10	0.500003	0.499997	
11	0.476194	0.523806	
12	0.454549	0.545451	
13	0.434786	0.565214	
14	0.41667	0.58333	
15	0.400004	0.599996	
16	0.384619	0.615381	
17	0.370374	0.629626	
18	0.357147	0.642853	
19	0.344831	0.655169	
20	0.333337	0.666663	
21	0.322584	0.677416	
22	0.312504	0.687496	
23	0.303035	0.696965	
24	0.294122	0.705878	
25	0.285718	0.714282	
26	0.277782	0.722218	
27	0.270274	0.729726	
28	0.263162	0.736838	
29	0.256414	0.743586	

Figure 46: Example of tabular viewer.

- Tabular, a tab will be used.

The default symbol is the comma. To export the current view as an image format, follow these steps:

1. From the *Export list*, select *Image Export*.
2. Click on the *Export* button.
3. You will be asked to enter a valid file name and select the *File type*.
4. After you finished entering the required information, click *save*.
5. The file is exported into the specified image format.

Valid image formats are: bitmap, gif, png, and jpg.

#### 6.5.4 Adding New Views

In addition to the default view, you can add many other views. Each one can be concerned with exploring the model from a different perspective. This can be done by first adding a new *view* and then determining the associated places or transitions. To add a new view:

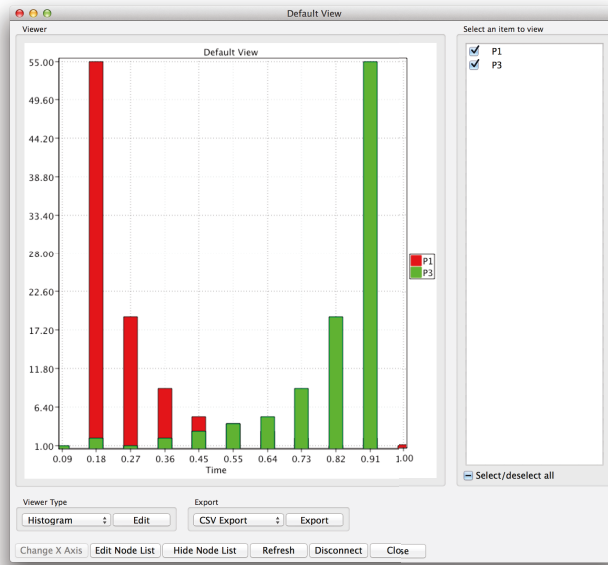


Figure 47: Example of histogram plot.

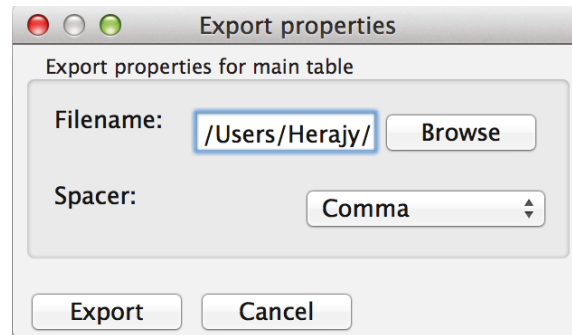


Figure 48: The export properties dialog.

1. In the list of buttons below the view, click *Add*.
2. Write a valid name for this view and click *Ok*, as shown in Figure 49.
3. The new view will be created and you can customise it as discussed earlier in this section.

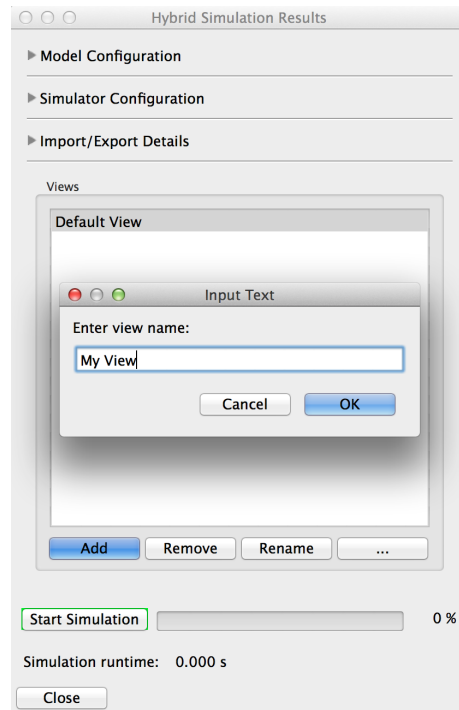


Figure 49: Creating a new view.

### 6.5.5 Removing Existing Views

If you have a *view* that you no longer need, you can remove it. To remove a view, select it and then click *remove* from the list of buttons.

### 6.5.6 Edit Viewer's Properties

When a viewer is selected, it loads its properties from the current view. These properties can be customised individually for each view. The number and type of these properties depend on the viewer type (i.e., tabular, XY-plot, or histogram). To edit the properties of a viewer, do the following steps:

1. On the view dialog and in front of the *viewer type* , select edit.
2. A multi-tab dialog will open as illustrated in Figure 50.
3. The number of tabs and properties will be different from one viewer to another.

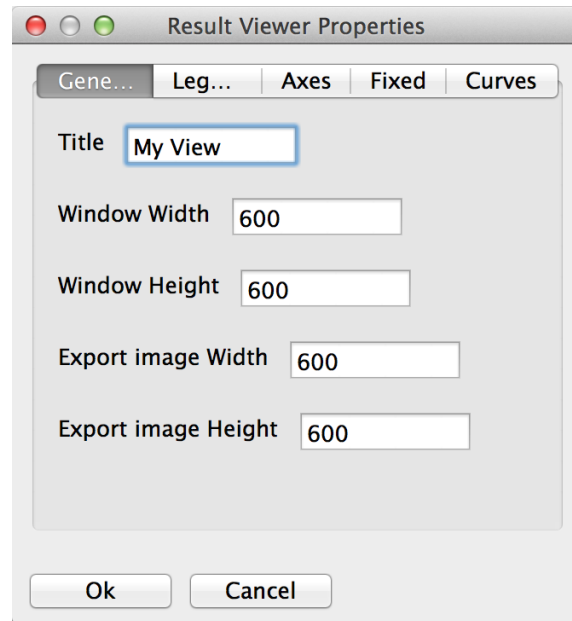


Figure 50: Adjusting viewer properties.

4. Adjust the properties and click Ok.

In the following, we list some of these properties and their meanings.

**Title** used to give a title to the view. This title is rendered on the viewer window. A view title can be set differently from a view name. New views get the view name as title. The title field applies to all the viewer types.

**Legend Tab** Applies for XY-plot and histogram plot.

- **Show legend** Switch on and off the legend.
- **Horizontal position** Control the vertical position of a legend. Possible values are: right, centre, and left.
- **Vertical position**  
Control the horizontal position of a legend. Possible values are top, centre, and bottom.

- **Sort curves**

Sort alphabetically the curve names ( names of the nodes selected) in the legend position. Possible values are: ascending (sort the curve names from A to Z), descending (sort the curve name from Z to A), and not sorted (use the order in which the user checks the curves).

**Axes Tab** Applies to the XY-plot and Histogram plot.

- **Show X-axis**

Switch on and off showing the X-axis label.

- **X-axis label** Set the label of the X-axis
- **Show Y-axis** Switch on and off showing the Y-axis label.
- **Y-axis label** Set the label of the Y-axis.

**Fixed Tab** Applies to the XY-plot and Histogram plot.

- **Fixed X-axis adjustment** Switch on and off using a fixed interval for the X-axis.
- **Fixed Y-axis adjustment**  
Switch on and off using a fixed interval for the Y-axis.
- **Min. X value** Set the minimum value of the X-axis; the default is zero.
- **Max. X value** Set the maximum value of the X-axis, the default is one.
- **Min. Y value** Set the minimum value of the Y-axis; the default is zero.
- **Max. Y value** Set the maximum value of the Y-axis; the default is one.

**Curves Tab** Applies to the XY-plot.

- **Show lines** Switch on/off drawing the curves as lines.
- **Show symbols** Switch on/off drawing the curves as symbols.
- **Line width** Set the width of the line curve.
- **Line style** Set the style of the line curve.



**Bar Tab** Applies to the Histogram plot.

- **Bar width** Set the width of each bar.
- **Interval width** set the discretisation interval.
- **Frequency** Select the type of the histogram. Possible values are count or percentage.

## 7 Export and Import

An  $\mathcal{HPN}$  model can be exported to and imported from different formats, including other Petri net classes. In this Section, we show different methods to export your constructed models as well as how to import models constructed using other tools.

### 7.1 Export to Other Petri Net Classes

To export  $\mathcal{HPN}$  models into other Petri net classes supported by Snoopy, follow these steps:

1. From the file menu, select *Export*.
2. The export dialog will appear as shown in Figure 51.
3. Use the check box to select the class(es) you would like to export to, then click *Export*.
4. The *Export Properties* dialog will appear as shown in Figure 52.
5. Select an appropriate file name and hit *Ok*.
6. The file will be exported with a message in the *log* window displaying the export details.

Valid Petri net classes that you can export your  $\mathcal{HPN}$  file to are: stochastic Petri nets, continuous Petri nets, in addition to coloured/uncoloured hybrid models. When exporting an  $\mathcal{HPN}$  into stochastic Petri nets, all the continuous components will be converted into discrete parts. On the contrary, when converting an  $\mathcal{HPN}$  into continuous Petri nets, all the discrete components will be converted into discrete parts. Moreover,  $\mathcal{HPN}$  can be exported into  $\mathcal{HPN}^C$  ones and vice versa. In the former, a simple colouring using the *dot* notation is performed, while in the latter, the net is unfolded. Please note that when exporting to other nets some information might be lost due to the conversion process.

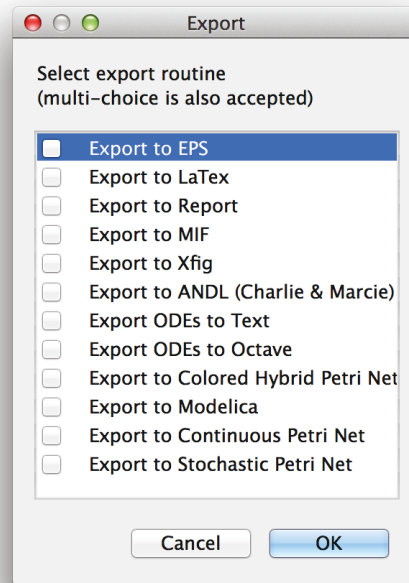


Figure 51: The export dialog box.

## 7.2 Export to (C)ANDL File

*ANDL* is an internal language used by Snoopy and our other tools to read and write low level Petri nets, while *CANDL* is used by coloured Petri nets. Moreover, advanced users can define their Petri nets using *(C)ANDL*.

To export (coloured) hybrid Petri nets to *(C)ANDL*, follow these steps:

- While opening the Petri net file, select *Export* from *File*.
- Select *Export to (C)ANDL* and then press *ok*.
- The *Export properties* dialog will open as shown in Figure 52.
- Select the file path and click *ok*.
- A text file will be created which you can open and read with a text editor of your choice.

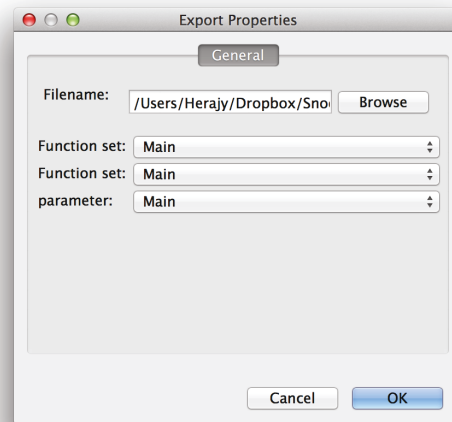


Figure 52: The export properties dialog box for  $(C)ANDL$  files.

### 7.3 Export as a Latex Report

For the purpose of documenting the Petri nets or publishing the final model specification, Snoopy permits generating a comprehensive report in the typesetting system LaTeX which includes all the model details. To export the Petri net file to Latex report, follow these steps:

- From *File* menu, select *Export*.
- Select *Export to Report* from the *Export* dialog.
- The *Export Properties* dialog will appear as shown in Figure 53.
- Select the appropriate properties, then click *ok*.

There are many options in this dialog which you can choose from. We are not able to cover all options in more details in this manual. For a detailed descriptions of all options related to exporting to a latex report, please refer to [Sha15].

### 7.4 Export to Other File Formats

In addition to exporting  $HPN$  models to other Snoopy net classes, you can also export the  $HPN$  models to other tools and other formats. Figure 51 lists the other file formats

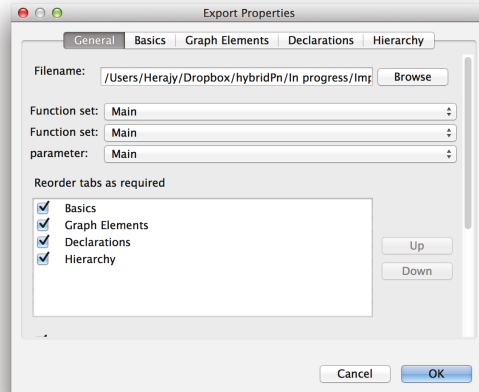


Figure 53: The export properties dialog box for latex reports.

that you can export to.

#### 7.4.1 Export to ODE

The system of ODEs representing the continuous part can be exported as a text file. To make such an export, follow these steps:

1. In the *Simulation Dialog*, click the collapse control called *Import/Export Details*.
2. The import/export section will appear as in Figure 54.
3. Click on the button *Save ODE*.
4. The *save as* dialog will appear asking for a valid file name.
5. Locate and enter a file name, then click save.
6. The system of ODEs will be exported as shown in Figure 55.

This option is useful when debugging  $\mathcal{HPN}$  models as well as to understand how Snoopy's hybrid simulator internally works. Please note that the set of exported ODEs consists of an ODE for each continuous place, in addition to an ODE representing the cumulative propensity which is used to carry out the switch between the stochastic and continuous regimes (see, [HH12] and [HH16]). The system of ODEs exported by this feature

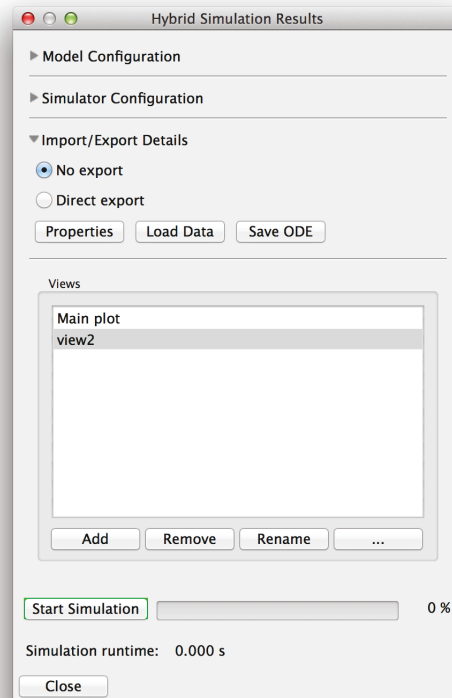



Figure 54: The simulation dialog with import/export details.

does not include ODEs due to discrete and stochastic components. For example, Figure 55 contains only one ODE representing the only continuous place in the net in Figure 5. The system of ODEs can also be exported to a latex file using a similar procedure to the one mentioned her.

## 7.5 Import

In some cases, you may already have a model and you would like to use it instead of constructing it from scratch. Snoopy provides you with the feature to import models available in other formats into one of the Petri net classes supported. Currently, you cannot perform a direct import to  $HPN$  or  $HPN^c$ . Instead you will need to import to stochastic Petri nets or continuous Petri nets and then export the model to  $HPN$ .

To import a model from an other format, follow these steps:



```

dStruct/dt = (1000*tem)-(1.99*Struct)

==== Total propensity ODE =====
da0/dt = (0.025*gen)+(0.25*tem)+(1*tem)*read(1,tem)+(7.5e-06*gen*Struct)

```

Figure 55: An example of exporting an *HPN* model into a set of ODEs.

1. From the *File* menu, select *Import*.
2. The *Import* dialog will appear as shown in Figure 56.
3. Select the format from which you would like to import, and then click *ok*.
4. A dialog box will open asking for the file name as well as additional information.
5. Select the file name and click *Open*.
6. Your file will be imported into Snoopy.

One of the useful formats that you can import from is Systems Biology Mark-up Language (*SBML*). Therefore, if you have a file written in *SBML*, you can easily import it into Snoopy.

## 8 Useful resources

There are many other useful resource that you can make use of to learn more about Petri nets in Snoopy.

- The book chapter in [BHM15] is an excellent starting point if you are not familiar with Petri nets in general or hybrid Petri nets in particular.

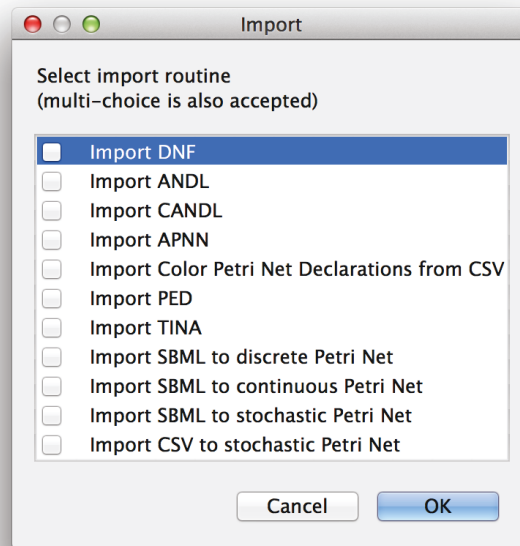


Figure 56: The import dialog.

- The Snoopy website <http://www-dssz.informatik.tu-cottbus.de/DSSZ/> contains plenty of models which you can explore to get an impression of how realistic models look like.
- If you are specifically interested in coloured hybrid Petri nets, then the coloured Petri net user manual [LHR12] will be very useful for you.
- Finally, there are many research papers where  $\mathcal{HPN}$  and  $\mathcal{HPN}^C$  have been discussed (e.g., [HH12, Her13, HSH13, HLR14]).
- Readers interested in the implementation of the simulation algorithms may want to check [HR02, HH12, HH16].

## 9 Frequently Asked Questions

1. **Q:** After installing Snoopy under MS Window I cannot open it. I got the following message: "missing msvcrt140.dll"

A: To fix this problem, please download and install the visual C++ redistributable for visual studio 2015 from this link: <https://www.microsoft.com/en-US/download/details.aspx?id=48145>. For more details about installing Snoopy, please refer to Section 3.

2. **Q: Why can I not find the *Start Simulation-Mode* in the *View* menu?**

A: First make sure that you have already opened a Snoopy file. Second, for qualitative Petri nets, it is not possible to use simulation, since such type of models can only be animated. However, if you have a qualitative Petri net, you can easily transform it into a hybrid model via export and import options available in Snoopy. For more details, please see Section 6.

3. **Q: I did not get any progress while simulating my model. What is the problem?**

A: This behaviour is usually due to a problem with the ODE solver. This can be caused by:

- **A problem with the initial step size:** try to use a smaller initial step. Depending on your model, this might be very small (e.g.,  $1e-20$ ).
- **A problem with the model structure:** currently, read and inhibitor arcs could cause discontinuity problems when they are used in the continuous part. Try to replace these arcs with modifier ones. Do not forget to replace the patterns with the actual formula if a modifier arc is used.

4. **Q: How to select the initial step size?**

A: For simple models, you can accept the default setting. If you face a problem, try smaller values for the initial step size until getting your model working. For more details, please see Section 6.

5. **Q: What is the best ODE solver for my model?**

A: You can use the default one: BDF. However, if your model is small ( $<100$  species), you can try the ARK solver. For more details, please see Section 6.4.2.

6. **Q: Which hybrid simulator to use for my model?**

A: The first answer: this is model-dependent. However, the best available simulator is the Improved HRSSA method. For more details, please see Section 6.4.3.



**7. Q: Can I use  $\mathcal{HPN}$  to draw and simulate stochastic Petri nets?**

*A: Yes, but it is advisory to select the stochastic simulator to get the best performance.*

**8. Q: When to use  $\mathcal{HPN}^C$  over  $\mathcal{HPN}$ ?**

*A: If your model exhibit structure repetition, then you should prefer  $\mathcal{HPN}^C$ . See Section 5.7 for more details.*

**9. Q: Do  $\mathcal{HPN}$  and  $\mathcal{HPN}^C$  use the same simulators?**

*A: Yes.*

**10. Q: When running the simulation for a time period from 0 to 1000, the simulation stops before reaching the 1000.**

*A: This means that something unusual happens during the numerical integration of the corresponding system of ODEs. You can find the exact cause by checking the log window. To overcome this problem, try to use a different ODE solver and/or a different linear solver.*

## References

- [Aga15] S Agarwal. Reengineering Snoopy’s GUI (Internship report). Technical report, Brandenburg University of Technology Cottbus, Department of Computer Science, July 2015.
- [BHM15] MA Blätke, M Heiner, and W Marwan. *BioModel Engineering with Petri Nets*, chapter 7, pages 141–193. Elsevier Inc., March 2015.
- [CVO] Sundials ccode website. <http://computation.llnl.gov/projects/sundials/sundials-software>. Accessed: 17/11/2016.
- [HBG<sup>+</sup>05] A Hindmarsh, P Brown, K Grant, S Lee, R Serban, D Shumaker, and C Woodward. Sundials: Suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Softw.*, 31:363–396, 2005.
- [Her13] M Herajy. *Computational Steering of Multi-Scale Biochemical Reaction Networks*. PhD thesis, Brandenburg University of Technology Cottbus - Computer Science Institute, 2013.
- [HH12] M Herajy and M Heiner. Hybrid representation and simulation of stiff biochemical networks. *J. Nonlinear Analysis: Hybrid Systems*, 6(4):942–959, November 2012.
- [HH15] M Herajy and M Heiner. Modeling and simulation of multi-scale environmental systems with Generalized Hybrid Petri Nets. *Frontiers in Environmental Science*, 3:53, 2015.
- [HH16] M Herajy and M Heiner. Accelerated simulation of hybrid biological models with quasi-disjoint deterministic and stochastic subnets. In E Cinquemani and A Donzé, editors, *Hybrid Systems Biology: 5th International Workshop, HSB 2016, Grenoble, France, October 20-21, 2016, Proceedings*, LNBI, pages 20–38. Springer International Publishing, 2016.
- [HHL<sup>+</sup>12] M Heiner, M Herajy, F Liu, C Rohr, and M Schwarick. Snoopy – a unifying Petri net tool. In Serge Haddad and Lucia Pomello, editors, *Proc. PETRI NETS 2012*, volume 7347 of *LNCS*, pages 398–407. Springer, 2012.

- [HLR14] M Herajy, F Liu, and C Rohr. Coloured hybrid Petri nets for systems biology. In *Proc. of the 5th International Workshop on Biological Processes & Petri Nets (BioPPN), satellite event of PETRI NETS 2014*, volume 1159 of *CEUR Workshop Proceedings*, pages 60–76. CEUR-WS.org, 2014.
- [HR02] E Haseltine and J Rawlings. Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. *J. Chem. Phys.*, 117(15):6959–6969, 2002.
- [HSH13] M Herajy, M Schwarick, and M Heiner. Hybrid Petri Nets for Modelling the Eukaryotic Cell Cycle. *ToPNoC VIII, LNCS 8100*, pages 123–141, 2013.
- [LH14] F Liu and M Heiner. *Petri Nets for Modeling and Analyzing Biochemical Reaction Networks*, chapter 9, pages 245–272. Springer, 2014.
- [LHR12] F Liu, M Heiner, and C Rohr. Manual for Colored Petri Nets in Snoopy. Technical Report 02-12, Brandenburg University of Technology Cottbus, Department of Computer Science, March 2012.
- [MPT16] L Marchetti, C Priami, and V H Thanh. HRSSA – efficient hybrid stochastic simulation for spatially homogeneous biochemical reaction networks. *Journal of Computational Physics*, 317:301 – 317, 2016.
- [Sha15] A Sharma. Snoopy Report Generator Snoopy2LATEX (Internship report). Technical report, Brandenburg University of Technology Cottbus, Department of Computer Science, July 2015.
- [SYSY02] R Srivastava, L You, J Summers, and J Yin. Stochastic vs. deterministic modeling of intracellular viral kinetics. *J. theor. Biol.*, 218(3):309–321, October 2002.