User's manual

http://www-dssz.informatik.tu-cottbus.de/DSSZ/
Fuzzy Petri nets and coloured fuzzy Petri nets in Snoopy
July 15, 2020

George Assaf, Monika Heiner and Fei Liu
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

List of Figures

List of Tables

Abbreviations

$\mathcal{CPN}$ Continuous Petri Networks

$\mathcal{FCPN}$ Fuzzy Continuous Petri Networks

$\mathcal{FCPN}^{\mathcal{C}}$ Coloured Fuzzy Continuous Petri Networks

$\mathcal{FHPN}$ Fuzzy Hybrid Petri Networks

$\mathcal{FHPN}^{\mathcal{C}}$ Coloured Fuzzy Hybrid Petri Networks

$\mathcal{FPN}$ Fuzzy Petri Networks

$\mathcal{FPN}^{\mathcal{C}}$ Coloured Fuzzy Petri Networks

$\mathcal{FSPN}$ Fuzzy Stochastic Petri Networks

$\mathcal{FSPN}^{\mathcal{C}}$ Coloured Fuzzy Stochastic Petri Networks

$\mathcal{PN}$ Petri Networks

$\mathcal{PN}^{\mathcal{C}}$ Coloured Petri Networks

$\mathcal{SPN}^{\mathcal{C}}$ Coloured Stochastic Petri Networks

$\mathcal{TFN}$ Triangular Fuzzy Number

This document explains the procedure of modelling and simulating $\mathcal{FPN}$ [1] and $\mathcal{FPN}^{\mathcal{C}}$ in *Snoopy*; please compare Figure 1 . Please note that the same steps for one net class can equally be applied to the other classes, just differentiate between uncoloured Petri nets ($\mathcal{PN}$) and coloured Petri nets ($\mathcal{PN}^{\mathcal{C}}$) [2].

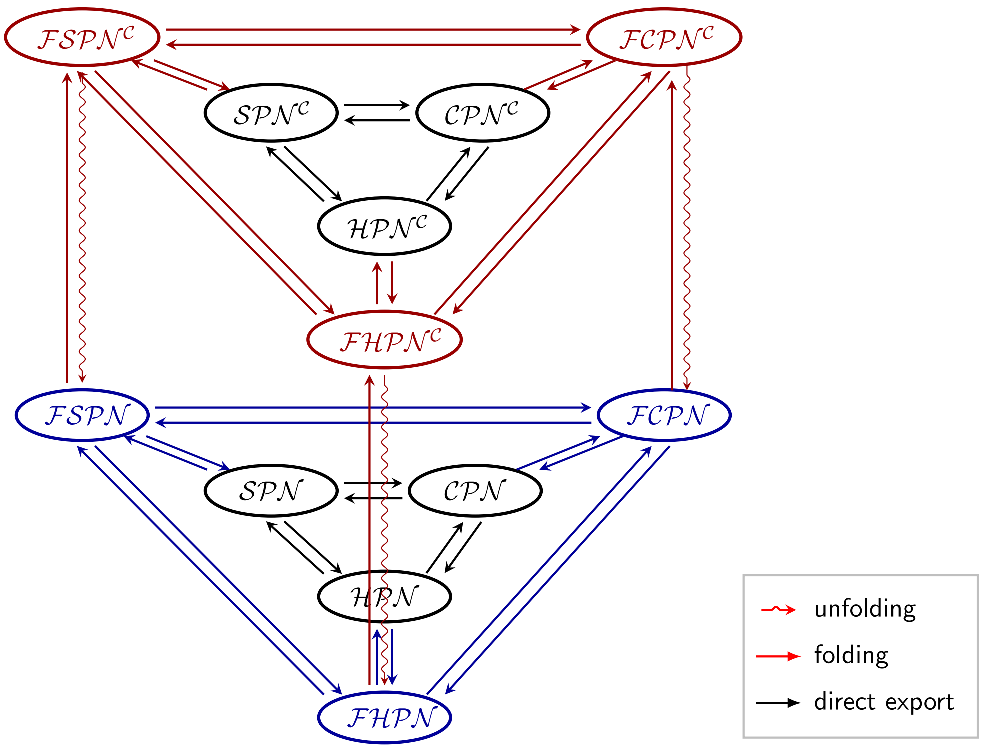

Figure 1: Export relation between some of *Snoopy*'s Petri net classes. Fuzzy nets differ from their crisp counterparts by additional pre-defined data types, supporting fuzzy numbers, which can be used as kinetic parameters. The extensions presented in [1] are coloured in blue, while the latest addition of net classes supported by Snoopy and their export relation are coloured in red. Please note that for clarity there are three folding/unfolding relations not shown in the figure ($\mathcal{SPN}$– $\mathcal{SPN}^{\mathcal{C}}$, $\mathcal{CPN}$– $\mathcal{CPN}^{\mathcal{C}}$, $\mathcal{HPN}$– $\mathcal{HPN}^{\mathcal{C}}$).

1 Fuzzy Petri nets (uncoloured)

There are three uncoloured $\mathcal{FPN}$ net classes, comprising $\mathcal{FSPN}$, $\mathcal{FCPN}$ and $\mathcal{FHPN}$. The modelling procedure of these classes can equally be applied. Please note that each net class has its own modelling elements; these elements are listed in the elements tree of that class.

1.1 Modelling

Creating an $\mathcal{FPN}$ model starts off with creating a new (empty) net file by choosing file menu and then clicking **new** command; the list of Snoopy net classes will appear as a result; compare Figure 3. The second way is to use the **Export** feature; compare Figure 2.

Figure 4 shows the fuzzy continuous Petri net model Decay dimerzation model which is adopted from [3] in which the transition r3 has k3 as a fuzzy kinetic parameter in its rate function; the Table 1 shows the rate functions associated with each transition.

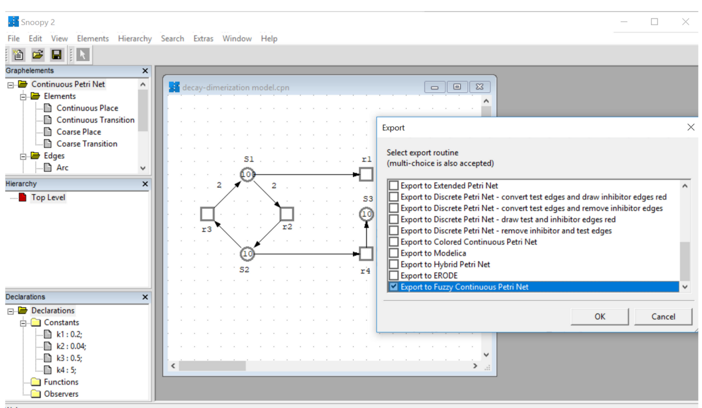

Figure 2: Creating an $\mathcal{FCPN}$ net by exporting a $\mathcal{CPN}$.

1.2 Constant definitions

Constant Definitions window allows users to create constants by specifying constants name, the group to witch the constant it belongs, the type and the corresponding value. $\mathcal{TFN}$ data type can be specified for defining a constant as a triangular fuzzy number; compare Figure 5.

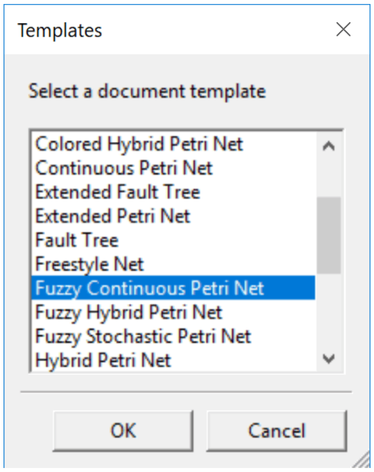

Figure 3: Creating a new (empty) $\mathcal{FCPN}$ model by selecting the appropriate template.

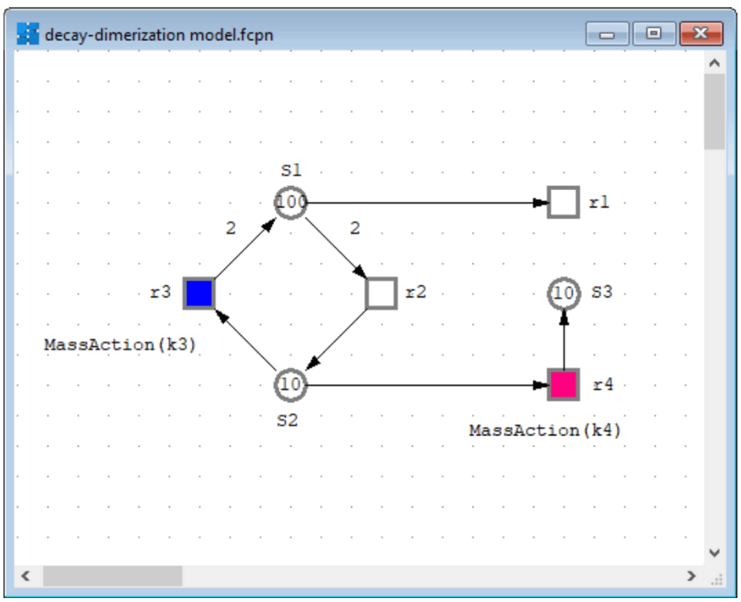

Figure 4: $\mathcal{FCPN}$ decay dimerisation model. The fuzzy kinetic parameters $k3$ and $k4$ are used in the highlighted transitions $r3$ and $r4$, respectively.

Table 1: Decay dimerisation $\mathcal{FCPN}$- rate functions of transitions, all following mass/action kinetics.

Transition r_i	Rate function	Kinetic constant k_i
r1	$k_1 \cdot S1$	0.2
r2	$k_2 \cdot S1 \cdot S1$	0.04
r3	$k_3 \cdot S2$	(0.45,0.5,0.55)
r4	$k_4 \cdot S2$	(4.9,5.0,5.4)

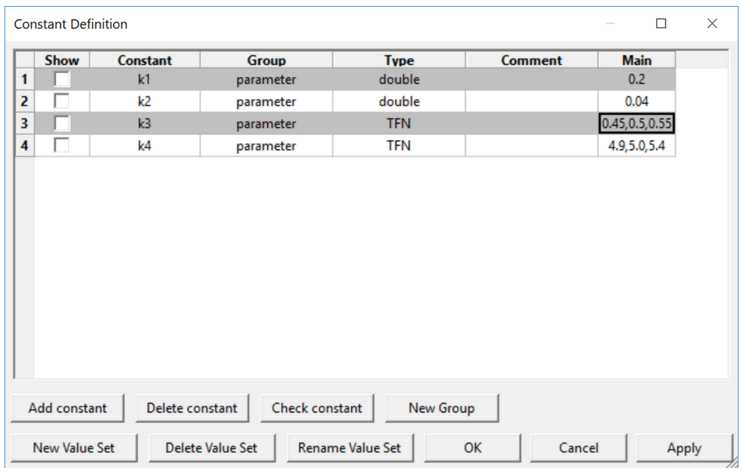

Figure 5: Constant definitions window.

1.3 Fuzzy Petri nets simulation

Simulation of $\mathcal{FPN}$ models starts off with switching to simulation mode (choose view Start simulation-mode from *Snoopy* ‘s view menu). Once simulation configuration window appears, a user can configure the experiment as usual. For $\mathcal{FPN}$ simulation settings, a user can specify number of alpha level, number of sampling points and the sampling strategy; compare FIgure 6. Then a user can start the simulation by clicking on start simulation button. Once simulation finishes, fuzzy band and membership functions of the selected variables can be viewed using viewer window; compare Figure 8.

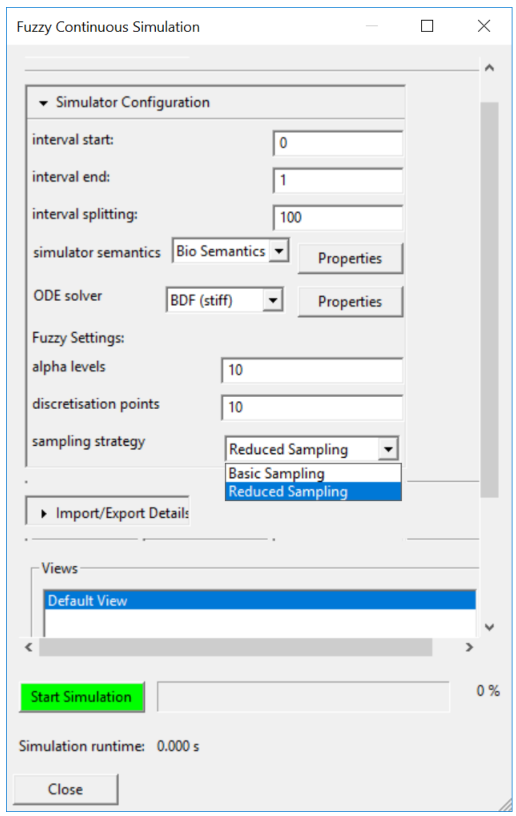

Figure 6: $\mathcal{FCPN}$ simulation configuration window. The simulation dialog consists of the same settings as the standard CPN simulation dialog.

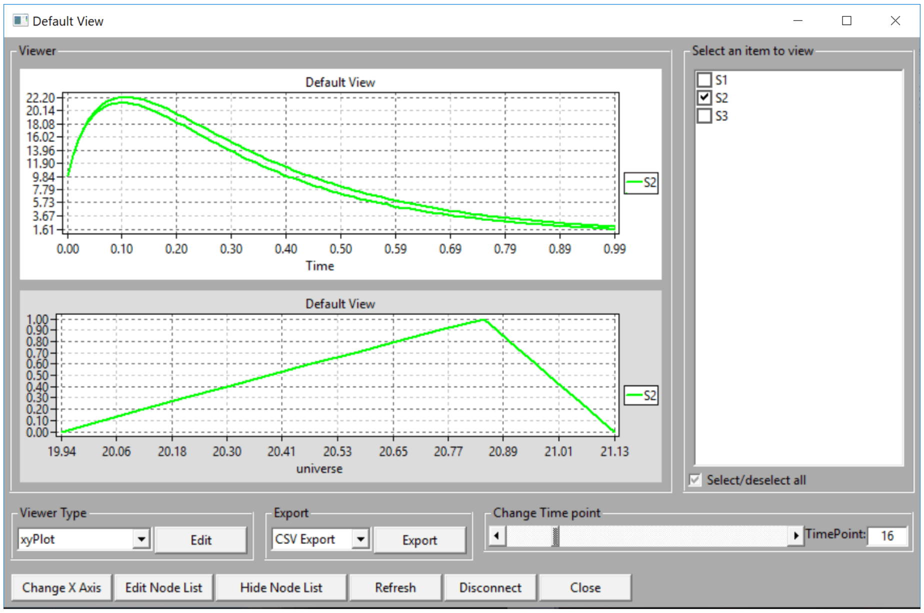

Figure 7: Simulation result viewer for the $\mathcal{FCPN}$ shown in Figure 4 and the kinetic parameters shown in Figure 5. The fuzzy band of species S2 is represented by the area located between the minimum and maximum curves.

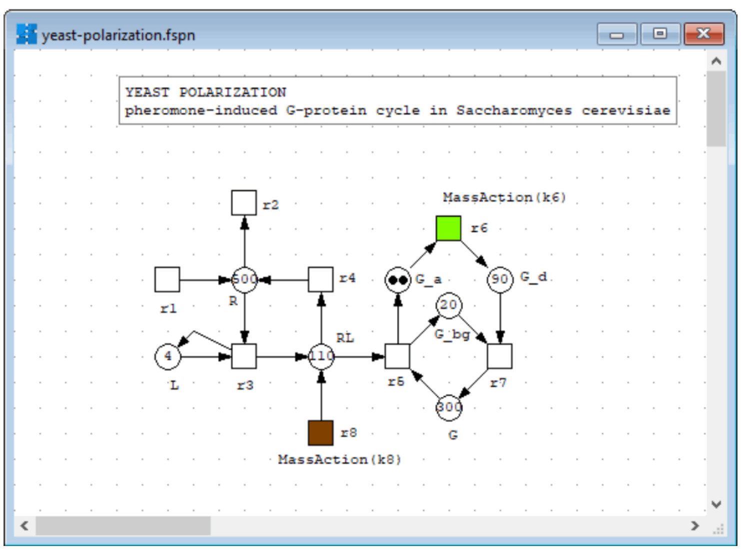

Figure 8: The $\mathcal{FSPN}$ model for yeast polarisation which is adopted from [5]. The fuzzy kinetic parameters $k6 = (0.08, 0.1, 0.12)$ and $k8 = (10, 13.21, 15)$ are used in the highlighted transitions $r6$ and $r8$, respectively.

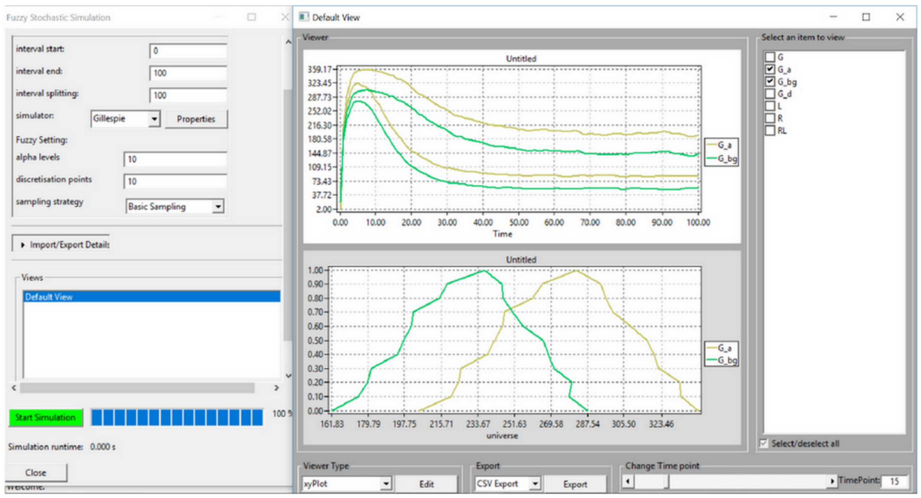

Figure 9: Screenshot of simulating the $\mathcal{FSPN}$ model (yeast polarisation), shown in Figure 8. The number of runs per each crisp stochastic simulation is set to 12. The two fuzzy bands of species G_a and G_bg are represented by the areas located between minimum and maximum yellow and green curves respectively.

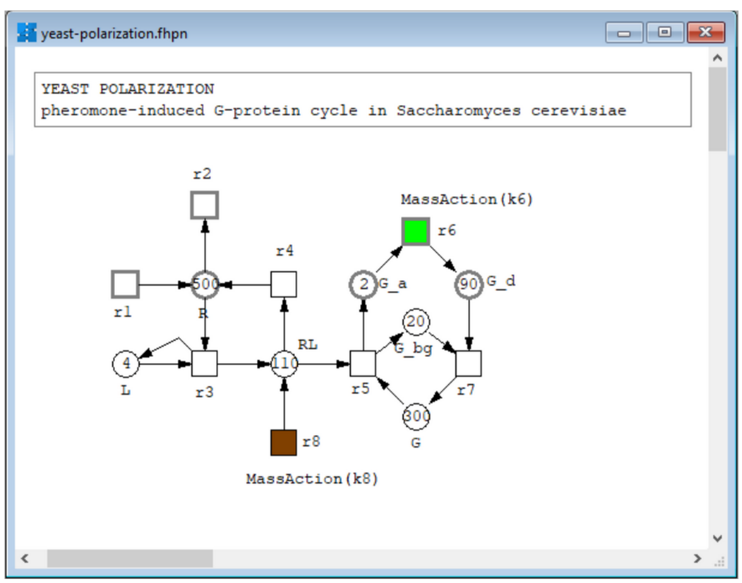

Figure 10: The $\mathcal{FHPN}$ model of yeast polarisation, obtained by exporting the $\mathcal{FSPN}$ model shown in Figure 8 to an $\mathcal{FHPN}$ model and converting a few nodes to continuous ones (drawn with thick grey line style)

.

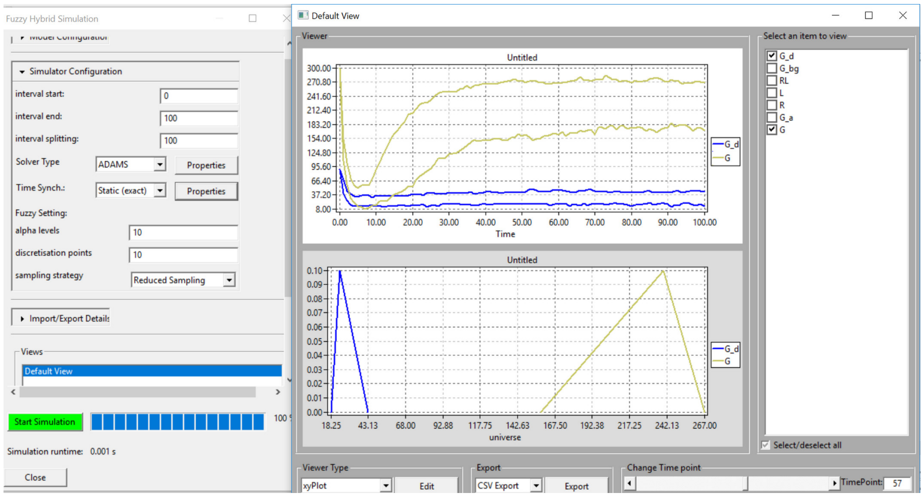

Figure 11: Screenshot of simulating the $\mathcal{FHPN}$ model (yeast polarisation), shown in Figure 10. The two fuzzy bands of species G_d and G are represented by the areas located between minimum and maximum blue and yellow curves respectively.

2 Coloured fuzzy Petri nets

Here we provide a collection of *Snoopy* screenshots outlining the workflow of modelling and simulating $\mathcal{FPN}^\mathcal{C}$, comprising the $\mathcal{FCPN}^\mathcal{C}$, $\mathcal{FSPN}^\mathcal{C}$ and $\mathcal{FHPN}^\mathcal{C}$. We demonstrate the modelling and simulating procedures using Repressilator test case as coloured fuzzy stochastic Petri net, the coloured Fuzzy continuous Petri net and coloured Fuzzy hybrid Petri net can be applied in an equivalent way.

2.1 Modelling

The modelling procedure starts off with creating a new coloured Fuzzy Petri net file, a net class can be chosen from the list of *Snoopy*'s Petri nets family; compare Figure 12.

Coloured fuzzy Petri nets can also be created by using Export feature e.g., coloured stochastic Petri nets $\mathcal{SPN}^\mathcal{C}$ cen be exported to coloured fuzzy stochastic Petri net.; compare Figure 13.

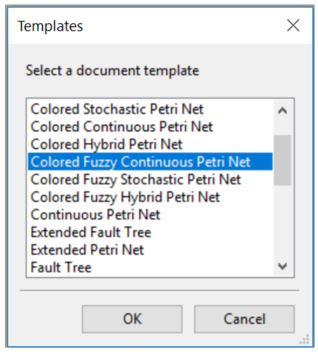

Figure 12: List of Petri net classes in *Snoopy*.

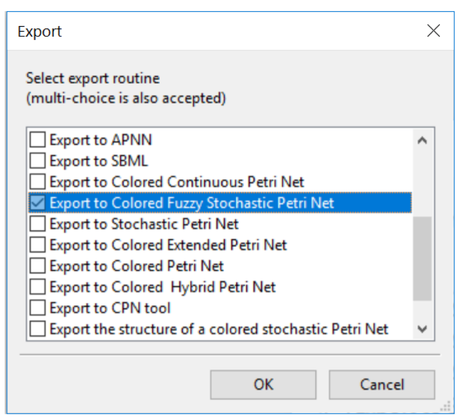

Figure 13: Export dialog of coloured stochastic Petri net.

2.2 Constant definitions

Once coloured fuzzy net has been created, a user can define constants using constant definitions window; by double clicking on the constants item on the declarations tree. A user can define constants to be used, e.g., in the net as whole or in the rate functions of transitions. For defining a constant as triangular fuzzy number, a user has to choose the $\mathcal{TFN}$ data type, then the value of the constants can be specified by writing the value directly in the value field; compare Figure 5, or by drawing it graphically using $\mathcal{TFN}$ drawing window which appears by double clicking on the value field ; compare Figure 14.

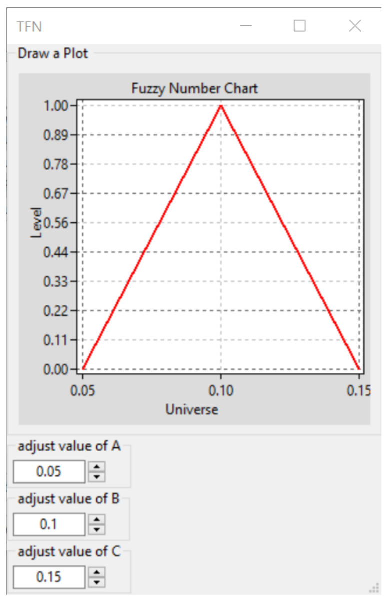

Figure 14: TFN drawing window.

2.3 Example - Repressilator

The coloured model of repressilator consists of three places and three transitions, each place is assigned to a colour set **Gene**, the transition **generate** has the following rate function

$$k\_gen * gene$$

, where k_gen is a fuzzy kinetic parameter $\mathcal{TFN}$ (0.05,0.1,0.15) and gene is a variable (pre-place); compare Figure 15. The Table 2 shows the transition rates associated to each transition.

The kinetic parameters can be colour-dependent,e.g., one colour can have a crisp kinetic parameter, and another colour a fuzzy kinetic parameter defined as

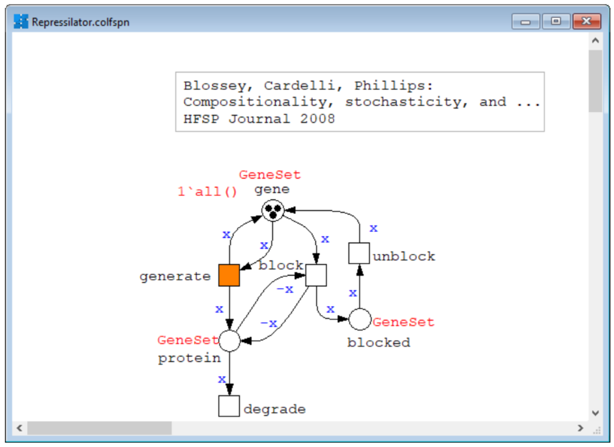

Figure 15: Coloured fuzzy stochastic Petri net model in *Snoopy* which is exported from $\mathcal{SPN}^{\mathcal{C}}$ [4]. The declarations: colorset GeneSet = enum with a,b,c, and variable x: GeneSet.

$\mathcal{TFN}$. In the Repressilator model, the coloured transition (in orange) indicates that this transition has a fuzzy kinetic parameter in its rate function, whereas the other transitions have crisp kinetic parameters (white).

2.4 Model simulation

Simulation of coloured fuzzy Petri nets requires unfolding step, by switching to the simulation mode, the unfolding engine appears; compare Figure 16.

After unfolding the coloured fuzzy stochastic Petri net, we implicitly get the unfolded fuzzy stochastic Petri net at the background, which is equivalent to the coloured model. As a result, unfolding the Repressilator gives: 9 uncoloured places, 12 stochastic transitions and 30 standard arcs. Furthermore, each transition instance will get assigned a function rate after evaluating the coloured function rate of its corresponding coloured transition, e.g., the function rate of transition **generate** will be evaluated to k_gen*gene_a, k_gen*gene_b and k_gen*gene_c.

Once the simulation result dialog appears, a user can configure the experiment settings e.g., simulation time, the simulator and its properties and the fuzzy-related setting e.g., number of alpha levels, number of discretisation points and sampling strategy, and then a user can start simulation by clicking Start Simulation button; compare Figure 17 for more details.

Table 2: Repressilator $\mathcal{FSPN}^{\mathcal{C}}$- rate functions of transitions.

Transition r_i	Rate function	Kinetic constant k_i
generate	$k\_gen \cdot gene$	(0.005,0.1,0.15)
degrade	$k\_deg \cdot protein$	0.001
blocked	$k\_block \cdot protein$	1
unblocked	$k\_unblock \cdot blocked$	0.0001

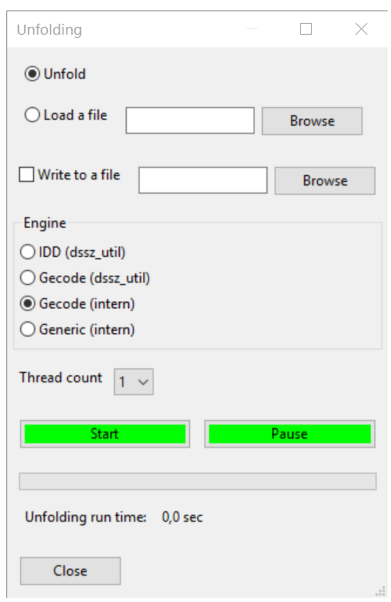

Figure 16: Unfolding engine dialog in *Snoopy*.

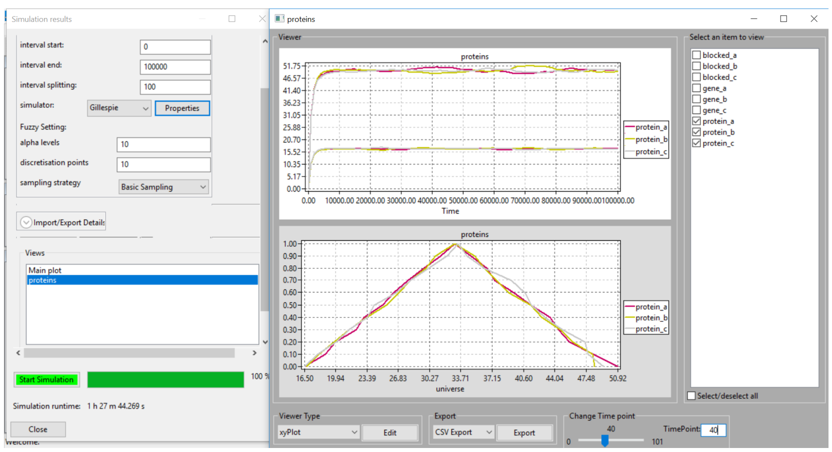

Figure 17: Result simulation Dialog in *Snoopy*; consisting of fuzzy band viewer and membership function viewer over time. Three curves are viewed, Protein_a, Protein_b and Protein_c, the membership function of the three curves at the time point 40. Please note that the number of simulation runs of Gillespie's simulator is set up to 10 000 runs.

References

[1] G Assaf, M Heiner, and F Liu. Biochemical reaction networks with fuzzy kinetic parameters in Snoopy. In L Bortolussi and G Sanguinetti, editors, *Proc. CMSB 2019*, volume 11773 of *LNCS/LNBI*, pages 302–307. Springer, September 2019. First Online: 17 September 2019.

[2] F. Liu. *Colored Petri Nets for Systems Biology*. PhD thesis, BTU Cottbus, Computer Science Institute, January 2012.

[3] F Liu, S Chen, M Heiner, and H Song. Modeling Biological Systems with Uncertain Kinetic Data Using Fuzzy Continuous Petri Nets. *BMC Systems Biology*, 12(Suppl 4):42, 2018. accepted for publication: January 2018.

[4] F Liu and M Heiner. *Petri Nets for Modeling and Analyzing Biochemical Reaction Networks*, chapter 9, pages 245–272. Springer, 2014.

[5] F. Liu, M. Heiner, and M. Yang. Fuzzy stochastic Petri nets for modelling biological systems with uncertain kinetic parameters. *PLoS ONE*, pages 1–19, 2016.