Spike - a command line tool for continuous, stochastic & hybrid simulation of (coloured) Petri nets

Jacek Chodak, Monika Heiner

Computer Science Institute, Brandenburg University of Technology
Postbox 10 13 44, 03013 Cottbus, Germany
jacek.chodak@b-tu.de, monika.heiner@b-tu.de
http://www-dssz.informatik.tu-cottbus.de

Abstract Spike is a command line tool for continuous, stochastic & hybrid simulation of (coloured) Petri nets (PN). It allows import from and export to various PN data formats. Its abilities comprise the manipulation of PN models by changing arc weights, markings or functions. It also unfolds coloured PN. To comply with the demand for reproducible simulation experiments, Spike is supported by a script language which allows for model and simulation configuration.

Keywords: continuous, stochastic, hybrid, coloured (hierarchical) Petri nets · simulation · configuration · reproducibility

1 Objectives

Many tools allow simulation of PN models. However, most of them have a graphical user interface (GUI) which usually involves additional dependencies and hinders batch processing. Simulation of PN models can be a time and memory consuming process. For performance reasons such simulations should be delegated to run on a server side. Due to the reasons above, GUI tools are not well suitable to be executed on a server.

Running simulation on a server helps to save user resources and speeds up simulations. On a server, the user can schedule multiple simulation experiments which can be executed simultaneously or sequentially. Often, a user wants to check how a model behaves for different sets of parameters. In this case, the user is forced to make changes in the model using an appropriate tool. Each time a model is changed, the simulation needs to be repeated. To compare how a model behaves under different types of simulation (stochastic, continuous, hybrid), it is necessary to configure, each time separately, the simulation and the model. This scenario can require to use separate tools for different types of simulations.

To ensure reproducible simulations, all parameters of model and simulation configurations have to be saved. To simplify the workflow, the configuration of the model and simulation should be supported by a script language, which allows easy modification of parameters.

Spike is meant for running different types of simulations of PN models. It is supported by configuration scripts which permit to configure the model as well as the simulation and to run sequentially multiple simulation experiments. Storing configurations in scripts allows Spike to reproduce simulations in a user friendly way.

2 Functionality

Spike is a slim, but powerful brother of Snoopy [2] - it is the latest addition to the PetriNuts family of tools for modelling, analysis and simulation with Petri nets, specifically tailored to the investigation of biochemical reaction networks. The main focus of Spike lays on efficient and reproducible simulation of PN models. Spike also offers import and export of various exchange data formats and some basic reduction of PN models.

Simulation Similar to Snoopy, Spike is capable to run three basic types of simulations: stochastic, continuous and hybrid, each comes with several algorithms. Simulation of coloured stochastic, continuous and hybrid PN models is supported by unfolding them automatically to uncoloured models.

A given model is simulated according to the specified simulation type, despite of place and transition types in the model. That means all places and transitions are converted to the appropriate type. For example if a user wants to run stochastic simulation on a continuous model, all places and transitions are converted to the stochastic type. Likewise for stochastic models to be simulated continuously, all stochastic transitions are converted to continuous type.

Simulation results can be saved in CSV files which can be used later for analysis and visualisation. They may comprise user-defined combinations of traces of place markings, transition rates, as well as observers (auxiliary variables).

Conversion Spike supports the following data formats of PN models:

- ANDL and CANDL - human readable formats for Petri nets and Coloured Petri nets, respectively, used internally by the PetriNuts framework,
- SBML (the Systems Biology Markup Language) - an XML-based representation format designed to exchange computational models of biological processes [4],
- PNML - an XML-based interchange format for Petri nets [6],
- ERODE - a tool for the evaluation and reduction of chemical reaction networks [1].

Table 1 shows the data format conversions currently supported by Spike.

Reduction Spike is also able reduce structurally the model, by pruning clean siphons and constant places. In both cases, clean siphons and constant places

Table 1. Data format conversions currently supported by Spike.

From	To
ANDL	PNML, ERODE
CANDL	ANDL, PNML, ERODE
SBML	ANDL, PNML, ERODE
ERODE	ANDL, PNML

can be calculated by Spike or loaded from a file. It is also possible to save results of the calculation to a file, which can be used later by Spike or for other analysis purposes.

Further reductions may be applied by converting a model to the ERODE format, if the model is to be read as ordinary differential equations (ODEs). Reductions of a model can have a significant impact on simulation time.

Reproducibility To comply with the demand for reproducible simulations, Spike reads a script which allows for model and simulation configuration. The structure of the script is easily readable for the user and does not require any special tools for editing: a simple text editor is enough.

The configuration script allows among others:

- definition of constants, e.g.:

```
constants: {
  // name of a group, see ANDL specification
  all: {
    /* if constant does not exist
    * then it will be created and
    * can be used in the configuration,
    * for example in defining a place marking
    */
    M: "D/2 + 1"
  }
}
```

- set marking for places, e.g.:

```
places: {
  // example of use of the newly created constant M
  P: "1000`(M,M)"
  P_2_2: 500
}
```

- definition of auxiliary variables (observers) which allow for extra measures by defining numerical functions; depending on the type of observer, it can be defined for places, transitions or simultaneously for places, transitions and constants, e.g.:

```
observers: {
  place: {
    OP01: {
      function: "P_1_1 + P_2_3"
    }
  }
}
```

- defining multiple simulation configurations, which permits to run multiple experiments for one model configuration;
- defining multiple exports of simulation results by use of regular expressions over the nodes of which the simulation traces are to be recorded; it is possible to combine the results of places, transitions and observers, coloured and uncoloured, in one file, e.g.:

```
export: {
  // Array of places to save,
  // including colored places like P
  // in this example (if empty, export all)
  places: ["P_1_1", "OP01", "Grid.*", "D", "P"]
  // Array of transitions to save,
  // including colored transitions
  transitions: ["t3_1_1_1_2", "t3", "t3" ]
  // Array of observers to save (if empty, export all)
  observers: ["M01", "OT01"]
  to: "sim01-file01.csv"
}
export: {
...
}
```

3 Architecture

Spike is written in C++, it is available for Linux, Mac and Windows. It has a modular structure, where the modules are basically decoupled from each other. This allows for easily adding new features.

Modules communicate with each other using command patterns and a queue of commands which is globally accessible. Each module has its own list of commands with specific parameters, which must be registered to the queue during initialisations of a module. Table 2 shows a summary of all commands currently available in Spike.

Commands are processed in a sequential way. Each command is executed by the module which is responsible for it. Let's consider the following use case illustrated in Fig. 1 – the execution of a simple configuration script. When the command "exe" is at the head of the command queue, the module Configure will execute it. During execution, the configuration module communicates with other modules by appending new commands to the queue.

Table 2. List of Spike's modules with their commands.

Module	Command	Description
Main	version	display version of Spike
CLI	help	display help for a given command
Configuration	exe	execute configuration script
Converter	load	load a model from a given file
	save	save a model to a given file
	prune	prune a model
	eval	evaluate constants and places
	unfold	unfold a coloured model
Simulation	sim	run a simulation of the model

4 Use cases

Spike permits to run simulations on a server as well as on the user side. It can be done in batch mode or by integration of Spike as a service. Algorithm 1 illustrates a typical scenario, which allows, e.g., to compare how a model behaves under different types of simulation algorithms or under different configurations of a given simulation algorithm.

The discussion of more scenarios exceeds the given space limit.

Algorithm 1: Use case to run multiple simulation configurations.

```
1 Load model
2 Set model configuration
3 Set simulation configurations
4 for each simulation configuration do
5 |   Run simulation
6 |   Save results of the simulation
7 end
```

5 Comparison with other tools

So far there is no tool, which allows to conveniently configure simulation experiments with support of a wide range of time-dependent Petri nets classes and simulation types. For comparison of Spike, two tools were chosen which provide partially similar functionality.

COPASI [3] supports stochastic and hybrid simulation of biochemical networks. It allows the definition of multiple result exports. However, there is no direct support for Petri nets. Configuration files follow an XML-based format, which hinders their readability by a user.

Renew (The Reference Net Workshop) [5] supports modelling and simulation. It permits simulation on a server [7]. However, the Renew core does not support quantitative (stochastic, continuous and hybrid) net classes.

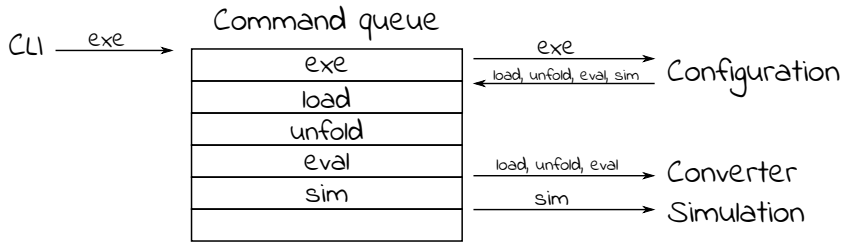

Figure 1. This example shows the flow of commands through the modules of Spike when a user types the command "exe".

To summarise, Spike is specifically suitable for scenarios, when user experiments require different configurations of a model and/or simulation.

6 Installation

Spike is available for Linux, Mac and Windows. Binaries can be downloaded from its website `http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Spike`. See also Spike's website for installation instruction and manual. Currently, Spike is under extensive development and we are open for suggestions.

Acknowledgement

Spike uses software libraries that have been developed by former staff members and numerous student projects at Brandenburg University of Technology, chair Data Structures and Software Dependability.

References

1. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: ERODE: A Tool for the Evaluation and Reduction of Ordinary Differential Equations. In: TACAS 20017. pp. 310–328. Springer, LNCS 10206 (2017)
2. Heiner, M., Herajy, M., Liu, F., Rohr, C., Schwarick, M.: Snoopy – A Unifying Petri Net Tool. In: ATPN 2012. pp. 398–407. Springer, LNCS 7347 (2012)
3. Hoops, S., Sahle, S., Gauges, R., Lee, C., Pahle, J., Simus, N., Singhal, M., Xu, L., Mendes, P., Kummer, U.: Copasi–a complex pathway simulator. Bioinformatics **22**(24), 3067–3074 (2006)
4. Hucka, M.: Systems biology markup language (sbml). Encyclopedia of Computational Neuroscience pp. 2943–2944 (2015)
5. Kummer, O., Wienberg, F., Duvigneau, M., Schumacher, J., Köhler, M., Moldt, D., Rölke, H., Valk, R.: An Extensible Editor and Simulation Engine for Petri Nets: Renew. In: ATPN 2004. pp. 484–493. Springer, LNCS 3099 (2004)
6. Petri Net Markup Language (PNML): Systems and software engineering – High-level Petri nets – Part 2: Transfer format (2009), ISO/IEC 15909–2:2011
7. Polasek, P., Janousek, V., Ceska, M.: Petri net simulation as a service. In: PNSE@ Petri Nets. pp. 353–362 (2014)