DiPS User Manual

December 29, 2019

Contents

Chapter 1

Introduction

Parameter inference of dynamical systems from experimental data is a difficult task. Measurements even from a single experiment may differ from one to another due to stochasticity of the studied system, measurement errors, or only partial information about the initial conditions. Estimation of parameter values representing such data becomes even more challenging task.

DiPS is a tool for data informed parameter estimation for parametric discrete time Markov chains (pMC). In the first step, parameter synthesis of pMCs from specifications expressed in Probabilistic Computation Tree Logic (PCTL) provides a symbolic representation of satisfaction in the form of rational functions. To do so we leverage existing tools PRISM [KNP11] and Storm [DJKV17]. Resulting rational functions, which represent the probability of satisfaction of the properties, are the cornerstone of the tool as all further analyses are based on them. In the next step, the (experimental) data serve as the observed probability of the PCTL specification. We use the data to constrain the parameter space of the rational functions; hence the name data informed parameter estimation. Finally, the points in which the rational functions are compliant with the data[1] are estimated. We employ space sampling deciding whether each function in sampled points is within the respective interval created from the data, space refinement splitting parameter space to obtain regions for which each rational function is within the respective interval, Bayesian inference, using Metropolis-Hastings, searching for the values which are most probable to represent the data, and optimisation, searching for function values with the least distance from the data.

[1] depending on the chosen method

Chapter 2

Installation

Currently, we support Windows and Linux distributions. If you are having trouble while using macOS, please contact us.

2.1 Download

DiPS can be obtained at `https://github.com/xhajnal/DiPS` as Python source code.

2.2 Dependencies

Before running DiPS, please install following dependencies:

- Python 3 `https://www.python.org/`
  - Windows - just Python
  - Ubuntu/Debian - Python header files should also be installed, please use `sudo apt install python3-dev`
  - Fedora/CentOS - Python header files should also be installed, please use `sudo dnf install python3-devel`
- PRISM 4.4[1] `http://www.prismmodelchecker.org/`
- tkinter - python library
  - Windows - already done

[1] newer or older versions are probably also compatible

  - otherwise follow instruction from `https://tkdocs.com/tutorial/install.html`

- other missing python packages - in the main directory run `pip3 install -v .`

To use the tool with interactive python environment, you can use Jupyter Notebooks - `https://jupyter.org/install`. To run parameter synthesis commands, which we provide for Storm, please install it - `http://www.stormchecker.org/`. Are you having trouble with z3 library? Read `README-z3.md` in the main directory. If you are still having trouble, please contact us.

2.2.1 Setup Config file

In the main folder, there is a configuration file - `config.ini`. Please insert location of PRISM to *prism_path* in *mandatory_paths*.

The rest of the config is optional; you can leave it blank. Change it to alter default paths to load/save files:

[mandatory_paths]

prism_path: path to bin folder in PRISM folder - `PRISM/bin`

cwd: path to the main folder eg. `MyDiPS`

[paths]

models: path to PRISM models eg. `MyDiPS/models`

properties: path to PRISM properties eg. `MyDiPS/properties`

data: path to data eg. `MyDiPS/data`

results: path to save results (all the results are saved in the subfolders) eg. `MyDiPS/results`

tmp: path to save temporal/intermediate files eg. `MyDiPS/tmp`

2.2.2 Run

GUI

To run GUI of the tool open command line in the main directory of the tool[2]. Then use the following commands.

[2]on Win - please open it with admin privileges to ensure changing the PRISM setting does not fail on permission denied

» cd src
» python gui.py

Graphical User Interface should appear now (With some output return in the command line).

Jupyter notebook

To run Jupyter notebook open command line in the main directory of the tool[3]

» cd ipython
» jupyter notebook

Several notebooks appear now:

- `create_models_and_properties` can be used to automatically create models, properties for population models presented in [HNŠP19].
- `synth_params` serves to synthesise parameters resulting with the rational functions
- `sample_n_visualise` samples and visualises result rational functions
- `generate_data` generate synthetic data by simulating the model
- `direct_param_synth` creates commands to be used for "direct" constrain solving using PRISM and Storm without deriving rational functions.
- `analysis` employs parameter space refinement using z3 solver or interval arithmetic to solve computed constraints

to follow the workflow of the paper [HNŠP19] just run the notebooks in this order. Documentation and source code of used functions is in `Mpm/src`. When you are familiar with the notebooks, try your input files or even adapt the notebooks.

[3] on Win - please open it with admin privileges to ensure changing the PRISM setting does not fail on permission denied

Chapter 3

Common information

3.1 Model Syntax

3.2 Properties Syntax

3.3 Results Output Format

Chapter 4

Graphical User Interface

4.1 GUI in General

The *graphical user interface* (GUI) consists of six parts - tabs.

4.2 Model & properties tab

Tab 1 - Model & properties allows to load and view DTMC model in `.p` format and PCTL properties in `.pctl` format.

4.3 Synthesise functions tab

Tab 2 - Synthesise functions serves to run the parameter synthesis of selected model and properties. The results are parsed by the *parser* and then can be factorised and stored as a pickled list in `.p` format. PRISM and Storm results or the pickled file can be loaded here directly.

4.4 Sample functions tab

Tab 3 - Sample functions tab visualises synthesised/loaded functions in a given parameter point or one-by-one point of the sampled grid. If the data are loaded, the respective value is visualised along with the value of the function, and if the data intervals are computed, the information whether the values of the functions are within the respective interval is shown.

4.5 Data & Intervals tab

Tab 4 - In *Data & Intervals*, data can be loaded, edited, and saved. At this point, the distance of synthesised/loaded functions and data can be optimised. The results, optimised parameter point, respective function values, and the distance, are shown as text and can be saved to a file. Also, the settings for computation of the intervals and the results are shown in this tab. Moreover, data-informed properties, which can be again used in PRISM or Storm, are produced after intervals are computed.

4.6 Constraints tab

Tab 5 - *Constraints*. In this tab, the inequalities combining the rational functions and lower/upper bounds of the intervals are obtained. List of constraints can also be pickled and loaded. Note that more general constraints can be loaded here.

4.7 Sample & Refine space tab

Tab 6 - *Sample & Refine space* is the main results part in which the results of space sampling, space refinement, and Metropolis-Hastingsare shown. The results can be saved and later loaded. A textual representation of the obtained space is updated after sampling or refinements finishes or after loading. Moreover, the GUI shows the loaded files above the mentioned tabs and also provides autosave of results in tmp folder.

Bibliography

[DJKV17] Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk. A STORM is coming: A modern probabilistic model checker. In *Computer Aided Verification*, pages 592–600. Springer, 2017.

[HNŠP19] Matej Hajnal, Morgane Nouvian, David Šafránek, and Tatjana Petrov. Data-informed parameter synthesis for population markov chains. In Milan Češka and Nicola Paoletti, editors, *Hybrid Systems Biology*, pages 147–164, Cham, 2019. Springer International Publishing.

[KNP11] Marta Kwiatkowska, Gethin Norman, and David Parker. Prism 4.0: Verification of probabilistic real-time systems. In *International conference on computer aided verification*, pages 585–591. Springer, 2011.